ADDIS ABABA UNIVERSITY
SCHOOL OF GRADUATE STUDIES
COLLEGE OF NATURAL SCIENCES
SCHOOL OF INFORMATION SCIENCE

AUTOMATIC PART-OF-SPEECH TAGGER FOR TIGRIGNA LANGUAGE USING HYBRID APPROACH

BY
MULUGETA ATSBAHA SIUM

A THESIS SUBMITED TO
THE SCHOOL OF GRADUATE STUDIES OF ADDIS ABABAUNIVERSITY IN PARTIAL FULFILMENTOF THE REQUIREMENT FORTHE DEGREE OF MASTER OF SCINECE IN INFORMATION SCIENCE

October, 2016

ADDIS ABABA UNIVERSITY

SCHOOL OF GRADUATE STUDIES

COLLEGE OF NATURAL SCIENCES

SCHOOL OF INFORMATION SCIENCE

AUTOMATIC PART-OF-SPEECH TAGGER FOR TIGRIGNA LANGUAGE USING HYBRID APPROACH

BY

MULUGETA ATSBAHA SIUM

Signature of the Board of Examiners for Approval

	Name	Signature	Date
1.	Solomon Tefera, Advisor (PHD)	_____	_____
2.	_____	_____	_____
3.	_____	_____	_____

Dedicated to:

1. **My Father Atsbaha Sium**

   Father I always remember all the suffer, pain and worst things that you pass to educate us, you expose yourself to such things which are unbelievable. Thanks dad for everything that you made perfects for all of us especially to me.

2. **My MotherAbraha GebreGzabihere**

   Mother all things that you made to me is perfect, I would like to say thanks only since I have no words and power to explain about you.

3. **My Brothers Kalayou Atsbaha**

   Kale, I have a lot of words to express peoples. However, none of them are enough to say things about you. Just, just, just thank you bro.

ACKNOWLEDGMENT

The completion of this thesis work would not have been possible without the support and inspiration from many people that have been influential by directly contributing to my research or by providing me unfailing encouragement and love in my personal life. Though, words fail in express my gratitude to everyone, I still want to say THANK YOU! I wish to express my gratitude to my advisor, Solomon Tefera (PHD), for his challenging and insightful comments on this thesis. I have learned a great deal of tagging and tagger implementation skills from him. He has helped me to rectify and correct every mistake. I would like to thank him for all the assistance, whether academic, technical or administrative, he gave me during my period of research. Needless to say, all errors that remain in the text are my own.

Next to this my thanks go to Tekly Gebregzabiher for his kindness and guidance on how to start my aim. To my friends, thanks for the entire moral supports that they had given me and many others. Individuals that I am unable to list but have directly or indirectly help me in completion of my thesis.

Tesgay Weldemariam I can't express my love and my thanks through words and sentences, exactly you act as my brother and father you give me a lot of experience and knowledge no one give me before , and you always give me moral and courage to proud . Tsegay thanks so much!

Hafte Abera: Dear Hafte there are a lot of words which have the power to express many thanks to person that made a great impact on life of someone, But you made a lot of things to make me correct and show me who I can finish my work in the required manner. None of the words is found to express my feelings to you. Thanks Hafte for everything you made.

Kalayou Atsbaha: there is nothing to say about you!!! I wrote these words even straggle with my tire, simply please THANK!!! Proud to be your brother! I would like to express my gratitude to my beloved family for their encouragement, patience, and their support they have given me during the course of this thesis.

Last but not least, I would like to thank to those who made me to the things that I want to start from high school till now.

Mulugeta Atsbaha Sium

2016

Table of Contents

List Of Figures

List Of Tables

Acronyms and Abbreviations

AffixT: affix Tagger

ANN: Artificial Neural Network

ANN: Artificial Neural Network

CRF: Conditional Random Field

CRF++ : Conditional Random Filed

HMM: Hidden Markov Model

MaxEnt: Maximum Entropy Model

ME: Maximum Entropy

MEMMs: Maximum Entropy Markov Modeling

MLP: Multi layer Perceptron

NLP: Natural Language Processing

NLTK: Natural Language Processing Tool Kit

OAVGPT: Original Averaged Perceptron Tagger

MAVGPT: Modified Averaged Perceptron Tagger

PoS: Part of Speech

RegT: regular expression Tagger

RI: information retrieval

SVM: support vector machines

TCi: Temporary Corpus at iteration 'i' ranges (1-n)

TEL: Transformation-based Error-Driven Learning

TGC: Tigrigna Corpus

TnT: Trigram n Tagger

Abstract

Tagging is a process of associating word class categories markers for corpora contents as additional information. Tagging can be used as pre-processing step for other high level language technology applications, such as to develop stemming algorithm, to prepare annotated corpora, etc. The process of tagging is a challenging task with Tigrigna because of the nature and morphological complexity of the language, resources scarcity and compiling Tigrigna texts.

The study uses a corpus containing 3100 sentences, 10000 distinct words and 56,151 total tokens and they are balanced corpus (not a domain specific corpus). A total of 22 Morpho-Syntactic course-grained tag-sets were adapted to prepare the annotated corpus using semi-supervised approach. Because the corpus is normalized, processed and annotated corpora it can be used for other language processing tasks.

The entire work describes an experimental study for improving Tigrigna tagger performance by combining outputs of two sequence taggers. Rule based, averaged perceptron taggers, and hybrid of the two taggers are investigated. The hybrid tagger was constructed from the sequence of the two taggers as averaged perceptron tagger followed by rule based tagger. The models are trained in 75% of the corpus and tested on the remaining 25% for their robustness and effectiveness. For each model several different experiments have been conducted.

Experimental result shows that reasonable tagger is achieved with modified rule based tagger along to three combined initial state annotator. In this study state-of-the-art tagging accuracy for morphological rich languages particularly Tigrigna with Averaged perceptron tagger is achieved. The Rule based tagger has found 94.8%, while Averaged perceptron tagger achieved 95.5%. Thus, averaged perceptron tagger and rule based tagger achieved comparable performance; however, the hybrid tagger improves the accuracy to 96.3%. The hybrid tagger works as a sequence of averaged perceptron followed with rule based tagger as error detection and correction sequence. In between the trained averaged perceptron and rule based tagger there is output analyzer with a threshold value as output validation and decision maker.

Therefore, the hybrid approach based rule based and averaged perceptron tagger creates a reasonable PoS tagger for Tigrigna.

CHAPTER ONE

INTRODUCTION

1.1 Background

Natural language processing (NLP) is a field of computer science that deals with an interaction among computer and human using natural language. The main issue is to improve human to human communication and human to computer communication (James & Daniel, 2006). It also involves designing applications which are able to understand and imitate natural language like speech recognition, speech synthesis, part of speech tagging, text summarization, discourse analysis, automatic summarization (Mirzanur, Sufal, & Utpal, 2009). However, unlike human it is highly complicated and complex for computer to understand natural language. There are different tasks that have the ability to process human language like spell checker, PoS, grammar checker, parser etc…. Among them part of speech tagging can be used as an early step text processing task for most other high level language processing task and IR applications. Part of speech tagging refers to associating word class markers to sentence members in particular and attaching additional information about corpora contents in the forms of tag label (Gebregzabiher, 2010; Mekuria, 2013). The task of PoS tagging can be used as an initial early steps in addressing more complex problems. Tagging provides significant information about a word and its neighbor, used for stemming in information retrieval, and parsing in NLP. Because, knowing word class of the candidate token helps to know morphological affixes that can be truncated (James &Daniel, 2006). Currently, there are different PoS taggers proposed for different language internationally and local languages like Amharic (Asress, 2008), Tigrigna (Gebregzabiher, 2010), Kafi-noonoo (Mekuria, 2013), Afan Oromo(Nedjo & Liu, 2014) and Sidamo (Addisu, 2016).

Language is a highly structured system of communication (Tewelde, 2002). Language plays a significant role in human life to facilitate day to day activity. Language in the written form keeps information and knowledge from generation to generation, in its spoken form it is used as communication channel (Mirzanur, Sufal, & Utpal, 2009).

There are several numbers of human languages in the world English, European and Asian, the same is true in Ethiopia like Amharic, Afan Oromo, Tigrigna and so on (Tewelde, 2002). Tigrigna uses inflectional and derivational morphologies to construct a word, thus, a given root word can result in a very large numbers of variant forms (Tewelde, 2002).

Considering resources limitation there has been almost no computational work on language processing tasks for Tigrigna. Thus, the researchers believe that as a crucial early step in information retrieval (IR) and other high level natural language processing (NLP) applications, designing PoS tagger for Tigrigna with different approaches ranging from simple to Complex PoS tagger is important.

Tigrigna requires robust and effective part of speech tagger for effective language processing and information retrieval applications. The first attempt on PoS tagger for Tigrigna has conducted by Gebregzabiher (2010); and the researcher use a hybrid of (HMM and rule based) to design the tagger which are a data driving taggers.

This paper focuses on investigating part of speech tagger using a hybrid approach that can be easily customized and uses Tigrigna features like morphological property. To make advantage of one tagger as input to the other, combining perceptron and rule based tagger is essential because both taggers have their own pros and cons. The sequence of tagger combination was adapted as a best practice from works of Asress (2008), Gebregzabiher (2010) and Mekuria (2013).

The main reason behind selection both taggers is they are an open-source available in the NLTK module. Specifically, Averaged perceptron tagger is simple, easy, and robust, can be implemented and trained with small and large amount of dataset. Averaged Perceptron tagger uses sophisticated features function that can be easily fit and customized with the nature and morphological complexity of Tigrigna. It has two options to assign tag that can easily fit Tigrigna features which is not in the HMM or other approach. Unlike the HMM tagger averaged perceptron tagger uses weighted average score values. Second averaged perceptron uses a dictionary lookup model to assign tag found in frequent word/tag dictionary without further assessment. Averaged perceptron training time is also relatively short compared to other data-driven methods when using a large training dataset (Loftsson & Östling, 2013). Averaged perceptron in Stagger has been

shown the state-of-the-art results to Icelandic language (Loftsson & Östling, 2013). Averaged perceptron in the NLTK toolkit with python can be easily managed and encoded.

In general averaged perceptron tagger is simplicity, short training times, an apparent room for improvement in (substantially) growing data size and can be easily customized to Tigrigna nature and morphology. The main advantage of rule based approach are, context of a language can be computed without language expert interference, it uses interpretable rules learned during the learning stage and rules can be simply interpretable. Rule based tagger uses a large amount of templates and it takes significant amount of time to train it, but it takes a fraction of memory size as compared to the HMM tagger(Sources). To overcome the bottleneck averaged perceptron tagger and Brill tagger, combination is as important as in the above sequence.

1.2 Statement of the Problem and Justification

Several NLP tasks were proposed to international and local languages. However, only limited works were conducted for Tigrigna, like Speech Recognition (Hafte, 2009), Part of speech tagger (Gebregzabiher, 2010), Tigrigna Applicative in Lexical Functional Grammar (Nazareth, 2011), and Phonetic based keyboard mapping (Okbeab, 2014). Among these, tagging is an NLP task used as preprocessor phase for other high level language processing researches (Gebregzabiher, 2010), (Mekurya, 2013). The main reason behind using tagging as a preprocessor phase for other high level language processing task are because it provides additional information about corpora contents in the form of word class category.

Different PoS taggers were designed for different languages including locally. However, Tigrigna receives little attention regarding tagging beyond the one developed PoS tagger 6 years ago. The developed Tagger is based on hybrid of Rule Based and HMM approaches. Even though, the tagger performance is good for the training and testing dataset the performance could be affected if the tagger is forced to tag new coming tokens because the tagger was trained for small corpus from a single source (Gebregzabiher, 2010). This indicates, the rule based tagger deduces only limited permissible rule. The HMM tagger lack probabilistic history affects significantly the probabilistic computation. Thus, performance of the hybrid tagger was affected (Gebregzabiher, 2010). According to Gebregzabiher (2010) par of speech tagger based on hybrid approach to Ti-

grigna was essential because performance of the tagger needs to improve. The new hybrid tagger was expected to minimize the gap by train the new hybrid tagger for large corpus in order to produce large permissible rules and to learn the morphology and context information as history for averaged perceptron tagger that can contribute for performance improvement of the hybrid tagger.

The existing hybrid tagger works at word level. The researchers believe that tagger on word level may have several constraints than tagger at sentence level specifically tagger for morphologically rich language like Tigrigna. Basically tagger at word level only uses candidate token meaning for tag assignment, because the tagging process is at word level possibility of using contextual information about candidate token is low compared to sentence level tagging. Thus, the tagger may lose the semantic meaning of the word; as a result tagging ambiguity is high. Thus, shifting the hybrid tagger that works at word level to sentence level is important.

As stated earlier different part of speech taggers research has been conducted for different language (Gebregzabiher, 2010; Mekuria, 2013; Addisu, 2016). Thus, the task of tagging remains as a hot research area generally for other languages and particularly for morphological rich languages like Tigrigna. Even if, there are approaching successfully implemented for other languages due to the variance in natures, morphological, syntactic behavior, and availability of resources, meaning of words and writing style it is impossible to implement model designed for one language to other.

On the other hand, even though, PoS tagger is designed for one language it is impossible to precede all possible sources of knowledge that can represent the language property and required for model development. Thus, investigating, designing and testing different PoS tagger based on different algorithm, methodology, taggers at different level for Tigrigna is essential. The researcher are also inspired to see how these techniques can be applied to Tigrigna which is totally and radically uses different morphology and syntax than the other languages, and what results can be obtained. Thus, this work uses a sequence of hybrid approach (averaged perceptron and rule based tagger) for Tigrigna as suggested by Gebregzabiher (2010). Besides combining the taggers we also inspired to see the effect of integrating different knowledge sources with individual taggers so that they can contribute to performance to the hybrid tagger.

Other high level language processing task requires corpora with additional linguistic information. Thus, it is obvious that corpora with PoS information are useful in many areas of linguistic and NLP applications. Therefore, even though, it is complex and expensive its wide range of application, because many words are ambiguous and its importance is not trivial (Hasan, UzZaman, & Khan, ND) makes PoS tagging important to design an automatic tagging.

Because only limited work was done for Tigrigna regarding information retrieval and NLP, many people uses Tigrigna as their main information processing face different problems; like no typographic error corrector application for both online and offline, lack of grammar checker and pronunciation dictionary. The researcher believes that the absence of well organized, accurate and efficient part of speech tagger for Tigrigna is considered as main obstacle for researchers to perform high level natural language processing applications. Therefore, it is the aim of this study to explore the possibility of designing sentence level Tigrigna part of speech tagger.

To this end, this study attempts to investigate and answer the following research questions.

- How to combine different source of knowledge to Rule based tagger to gain performance improvement of the hybrid tagger?
- How to determine and customize Averaged perceptron tagger parameters to improvement the performance of the tagger and that of the hybrid tagger
- How to process and prepare the corpus so that can contribute to performance improvement and determine the robustness of individual taggers and the hybrid tagger?

1.3 Objective of the Study

1.3.1 General Objective

The general objective of this study is to propose sentence level part of speech tagger using hybrid approach that improves the performance of Tigrigna tagger.

1.3.2 Specific Objectives

In order to achieve the general objective, the following specific objectives should be achieved:

- To study the structure of Tigrigna sentence
- To understand Tigrigna word class categories so as to identify and select/adopt the appropriate tag-set collection.
- To collect and prepare corpus for training and testing purpose
- To identify/select/adopt tag-sets that best suit the morphology of Tigrigna
- To investigate and explore an appropriate dataset for training and testing purposes
- To develop robust and effective PoS tagger for Tigrigna
- To test the performance of the proposed tagger

1.4 Scope and Limitation of the Study

Even the corpus is expected to cover the most important aspects of Tigrigna morphological property, however, information added to corpora content are used as to give course-grained information about word class category along with the course-grained tag-sets, but not about the issues like gender, number, and tense since they are expected to include in the low level fine-grained Morpho-Syntactic Tag-set and very-fine-grained Morpho-Syntactic tag-set.

Because there is no clear way of identifying Tigrigna compound words and abbreviation the study only use sample words as a look up table. Because the study examines affixes of Tigrigna language but not infix of words are also expected to examine with the aforementioned low level tag-sets.

1.5 Significance of the Study and Beneficiaries

PoS tagger Possess many advantages for different high level language processing tasks and can be used as a means of additional information attachment to corpora contents. Tagging can be used as a basement for developing information retrieval applications and other higher level NLP tasks including parsing, grammar checker and so on. Considering the importance of PoS tagger in addition to the previous work done the beneficiaries of the study is:

Peoples who want to conduct other high level NLP researches for Tigrigna such as Parsing, grammar checker, spell checking, Question Answering, intelligent applications, pronounce dictionary etc... It is also possible to researcher who wants to conduct with Stemming algorithm in

the field of information retrieval. Basically, for researcher who wants to extend the research using fine and very-fine Morpho-Syntactic tags can be used as a basement reference for their works. Besides that, linguistic researchers need to prepare annotated corpus with general information can be also the beneficiary of the study (Tag-set at course-grained level).

1.6 Methodology

In this study work we conduct experimental research towards improving performance of taggers by combining different approaches from a sequence order for Tigrigna. So the tools and techniques used to conducting the study are presented in the next section.

1.6.1 Literature Review

In this study different relevant published, unpublished text documents , books, journal articles, news, previous related works and electronic publication on the web have been consulted so as to

- Understand the process of part of speech tagging.
- Understand and to have clear general information about Tigrigna
- Identify and study different approach of part of speech tagging.
- Identify appropriate tools required to develop a prototype.

Moreover, to learn what others have done in the area and to better understand the problem, a comprehensive exploration of available empirical literature on PoS is conducted.

1.6.2 Data Collection Method

Different sources of Tigrigna text document were used to prepare Tigrigna corpora for this work. However, due to lack of Tigrigna texts the researcher prepares own corpus collected from different sources. Sources of the corpus include bible, text books, meeting, news, and reports which include agriculture, minerals, education, economy, religion, agenda, health, politics, interview, law, sport and so on. In addition to the above sources Tigrigna raw texts was also collected from different popular webs including Tigrigna newspapers called Aiga forum Ethiopia, Horn Africa

news, and Tigraionline. Therefore, to have common understanding about the collection we categorize the entire collection to three groups. First collection incorporates text collection for news paper. The collection covers different topics (Politics, economics, general, science, medical, sport, and art). The topics are varied so as to represent different morphological variation in Tigrigna. Statistically the corpus consists of 2923 distinct word 18730 total tokens and 1000 sentences. Second collection consist collection for bibles book. It consists of 2077 distinct words 10634 total words and 1100 sentences. Third collection consists of different sources like (Text book, websites, meeting, and agenda) from the selected sources by the researcher to represent the morphological complexity of Tigrigna. Collectively the collection consists of 5000 distinct words 26787 word tokens and 1000 total sentences. Statistical property of the corpus was present in Table 1. The researcher believes the collected raw text is representative of Tigrigna language nature. The researchers believe that, the collected text is balanced (corpus which is not Domain Specific). Even though, comparing size of the corpus to work did for other language is not as large enough, however, it is possible to say that, this is sufficient for this work.

Table1.1: Statistical property of Tigrigna Corpus

R/No	Corpus name	Total words	Unique words	Total sentence
1	Bible	10634	2077	1100
2	News	18730	2923	1000
3	Collection	26787	5000	1000
4	Total Distribution In The Corpus	56,151	10,000	3,100

Generally the task of collecting Tigrigna text corpora is quite challenging about at least three reasons. As mentioned earlier, the amount of information available on the web is limited to size and is variation. Second, random crawling with text does not serve the purpose of this research as the text's meta-data need proper documentation. Third, compiling Tigrigna raw text in to the form that is convenient for processing easily is also another big challenge.

1.6.3 Implementation Toolkit and Technique

The main tools used to design Tigrigna part of speech tagger are Natural Language ToolKit (NLTK) and Python programming language. NLTK is the most freely available tools for language processing. The rationale for choice of these tools is, NLTK is a collection of open source Python modules, linguistic data and documentation for research and development in natural language processing. It also provides excellent combination of hands-on access to data, explanation and real-life data. As mentioned NLTK is an open source freely available on the web and it contains python modules. NLTK provides a flexible framework of advanced projects and research. Some of them are developing a multi-component system by integrating and extending NLTK components; and adding on entirely new components. NLTK also provides standard implementation of all the basic data structures and algorithm, interface with standard corpora, large corpus samples, and a flexible and extensible architecture (Bird, Klein, & Loper, 2005). Python language is a simple and powerful scripting language with admirable functionality for processing linguistic data. Python is an interpreted language. So it is easy about just experiment - you don't need a complex development environment to use it. Most importantly Python is highly readable, object-oriented; in which each entity has certain defined attributes and methods (Bird, Klein, & Loper, 2005). Python can be downloaded for free from www.python.org.

1.6.4 Experimental Setup and Modeling

In this thesis work, three different approaches were investigated, these are Averaged perceptron tagger (Matthew, 2013), rule based tagger (Brill, 1995), and hybrid approach of (Averaged perceptron and rule based). The rule based approach as its name indicates to rely on rules which are either handcrafted or machine learned rules. Rules are a fundamental building blocks of the approach used to correct tag assigned by initial state annotator. Basically it uses lexical and contextual rules that automatically deduced from learning phase (Bril, 1995). The Transformation error driven learning approach was adapted to rule based tagger where both lexical and contextual rule are automatically learned from a manually annotated corpus. The second approach were averaged perceptron tagger as implemented by (Matthew, 2013), which is easy, fast, and works based on the concept of perceptron. The third approaches were designed by combining a sequence of two taggers (averaged perceptron, followed by rule based approach). This sequence works as er-

ror corrector and detector sequence. Initially, averaged perceptron tagger was placing then rule based tagger. After initial tag was assigned to candidate to-ken by averaged perceptron an expression was tested for the score provided by the averaged perceptron. Afterward, the expression will decide whether the tag is final or need further assessment (Mekuria, 2013). During hybrid the two taggers, up to the researchers knowledge and skill we try to integrate all necessary futures of the language in both taggers.

1.6.5 Tagger Evaluation

After developing the model for Averaged Perceptron, rule based and hybrid (averaged perceptron followed by rule based) tagger using different experiments its performance were evaluated to see the robustness and effectiveness. To evaluate the effectiveness of the tagger we use confusion matrix.

1.7 Thesis Organization

This thesis work was organized in to six chapters. Chapter one presents an overview and background of the study; Chapter two presents' literature review and related works on PoS tagger, Chapter three nature linguistic properties of Tigrigna sentences, word classes and tag-set selection Chapter four focuses on model investigation and design of PoS tagger for Tigrigna. Experiment and experimental result where discussed in Chapter five. Finally, conclusion and future works were presented in Chapter six.

CHAPTER TWO

LITERATURE REVIEW AND RELATED WORKS

2.1 Overview

There are several thousands of languages in the world and are classified into super-families and families. Tigrigna is a defacto national official language in Eritrea and regional official language in Tigray. Tigrigna belongs to Afro-asiatic super-family and Semitic family as like that of the Amharic and Gurage (Tewelde, 2002). Tigrigna is a morphological rich, highly complex and it suffers lack of NLP tools and application that process Tigrigna documents. This is due to scarce of Tigrigna resources and ready-made documented information used as a starting point to perform different language technology tasks (Tedla, Kazu-hide, & Ashuboda, 2016).

Language technology applications are applied to wide verity problems like business and administration to create superior and new useful solutions. In teaching and learning it is used to assist to disable; to carry out new services both to organizations and to clients. Natural language processing (NLP) is a language technology which deals with interaction among human and computer using natural language instead of computer language. The main ideal goal of NLP is to get computers perform useful tasks involving human language. It also involves designing of applications that have the ability to understand, and imitate natural language (Prakash et al, 2011).

NLP tasks plays significant role in solving different problem areas of natural languages, like speech recognition, speech synthesis, part of speech tagging, text summarization and so on. Part-of-speech (PoS) tagging is an important NLP task used to prepare annotated corpora (Gebregzabiher, 2010; Mekuria, 2013). The class marker can be (noun, verb, adjective, or whatever class marker). Output of the tagger can be used as pre-processing stage to other high level natural language tasks (Nedjo & Liu, 2014). Throughout this section we will explain in detail the methods and related works used as a starting point of this work.

2.2 Part of Speech Tagging

Part of speech tagging (PoS) is a well known task in the field of Natural Language Processing community (Mekuria, 2013; Nedjo & Liu, 2014). The process can be done either manually or automatically at word level or sentence level. Each method has its own pros and cons. For instance, if process was done at sentence level, whenever, error done for word located at the beginning of a sentence may propagate throughout the whole sentence. On the other hand if process was done at word level assuming context of a word and other syntactic property of a word might be difficult (Prakash et al., 2011). Manually tag assignment needs time, effort, and tedious. However, it produces more accurate result as compared to automatic tagging. Automatic tag assignment is simple and cost effective. Automatic tag assignment requires encoding features of language property to come up with well decision.

As explained (Gebregza-biher, 2010; Mekuria, 2013; Nedjo & Liu, 2014; Addisu, 2016) mostly the rule based and statistical based approach are preferred by different researchers. Rule-based taggers assign tag to candidate token using a predefined set of constraints, lexical and contextual information that are automatically generated rules from training set (Mekuria, 2013). The stochastic approach uses frequency, probability or statistical information to assign most appropriate tag sequence for a given sequence of words. Averaged perceptron is a new approach based on the concept of perceptron tagger (Matthew, 2013). It is a supervised machine learning algorithm used to associate word class information to corpora contents. Averaged perceptron uses a set of features and look up table to assign word class level categories on corpora contents.

For the last several year's combination (Hybrid model) of tagger is proposed to different international languages and local languages like Amharic (Asress, 2008), Tigrigna (Gebregzabiher, 2010; Kafi-noonoo, Mekuria, 2013). In all cases the main ideal logic in combining the tagger is to come up with better performance and to degrade limitation of one tagger with the next sequence of the second tagger.

Having this in mind, to the best of the researcher designing and investigation robust and effective PoS tagger for Tigrigna is essential. To do this, there are different programming languages which are freely available in the web used to process and scientifically manipulate natural language fea-

tures. Python programming language and NLTK are widely used tools (Gebregzabiher, 2010) and (Mekuria, 2013), which is freely available in the web and can be used freely. Python programming language is a language which contains interactive modules to process language text documents. On the other hand NLTK is natural language toolkit contains different python modules used to access and process natural language text document.

2.3 Approaches to PoS Tagging

As noted by (Gebregzabiher, 2010; Mekuria, 2013) several tagging approaches were proposed to different languages. According to their learning mechanism the process can be classified as supervised and unsupervised technique. Both differentiate in terms of their degree of tagging automation and training process.

Unsupervised tagging model doesn't require pre-tagged data rather it uses an advanced computational technique to induce the candidate sequence of tags for the candidate token. On the bases of the history they gain from the training (probabilistic information required by statistical models or transformation rule required by rule based tagger) they assign tag for unseen data (Das and Petrov, 2011).

Supervised learning is a task of inferring function of labeled training data. The training data contains set of training examples. Each example is a pair, consisting of an input object and a de-sired output value. The learning algorithm analyzes the training data and deduces an inferred function that can be used for mapping new examples. An optimal situation allows the algorithm to correctly determine class labels of unseen instances. This requires learning algorithm to generalize across training dataset to unseen dataset (Das and Petrov, 2011).

2.3.1 Rule Based Approach:

Automatic extracting linguistic information about text corpus can be a powerful method of overcoming linguistic knowledge acquisition bottlenecks that slowdown creation of robust and accurate natural language processing systems (Bril, 1995). To solve the problem several numbers of models was readily available and widely implemented for different languages. All models are trained and re-trainable on standard text corpora for different local and foreign languages (Ge-

bregzabiher, 2010; Mekuria, 2013; Nedjo & Liu, 2014). Thus, from the available tasks of associating word class marker to corpora contents in the form of word class label Rule based approach is the prominent one. Eric Brill introduced rules based PoS tagger in 1992 where grammar of the language is provoked directly from training corpus without the need of expert knowledge. However, the training corpuses which serve as input to the tagger were manually annotated corpus. According to Eric the tagger able to derive lexical and contextual information about the training corpus and 'learns how to infer part of speech tags for candidate token. Once the training is completed, the tagger can be used to annotate new corpora.

Rules are considered as a fundamental building block of the approach (Addisu, 2016). Rule based to approach relies on rules to assign PoS class marker for candidate token. Rule that were used to drive expected hit can be either handcrafted on traditional rule learning approach or ma-chine learned in which rule are automatically deduced during learning (Mekuria, 2013). Typical rule based approaches uses two kinds of rule derived from learning phase of the model. The rules are lexical rule which is directly related to the morphology of the candidate token and con-textual rules which are directly related to the context of the candidate token. These rules are of-ten known as context frame rules (Gebregzabiher, 2010), (Mekuria, 2013). As an example, a context frame rule might say: If an unknown word X is preceded by a determiner and followed by a noun, tag it as an adjective.

Dt - **X** - N = X/Aj

↓ ↓ ↓

እቲ ቀይሕ ወዲ።። (Ati qeyH wedi።።)

Most Rules based to approach requires supervised training (Mekuria, 2013). Thus, it requires an initial manually annotated data to automatically induce grammar information about the intended language. Otherwise if a rule is expected to deduce manually then linguistic expert on deep knowledge of the language is required to deduce the rules. Afterward, the deduced rules are used to associate word class label markers for unseen datasets.

Transformation-based Error-driven Learning is one of the most widely used rule based tagger that combines benefits of both rule based and probabilistic approach. Process in TEL, begins with un-annotated text as input passes through initial state annotator. Every candidate token was

assigned suggested word class label based on the initial state annotator and then, labeled dataset output from initial states annotator become a temporary corpus (TCo) and compared to the goal corpus. In each pass the learner produces one new single rule that improves the initial temporary corpus (TCo) as compared to the goal standard corpus. Each time the temporary corpus passes through it, and it replaces the temporary corpus (TCo, TC1…TCn) with the analysis results when the rule is applied to the new corpus. Through this process the learner produces a structured list of suggested rules. Each rule is suggested as a best rule based on the score threshold values limited in training phase for the rules to be bests. Accordingly, only the rule which got bests hit over the rest of all rule was applied for the temporary corpus. Then, rules are generated until there is no rule found to improve the TCi to TCi+1 which is expected to be the TCi+1 is closest to the gold standard corpus than the TCi.

The derived rule during learning phase consists of two parts condition (the trigger and possibly a current tag), and resulting tag. Rules are instantiated from a set of predefined transformation Brill templates. They contain un-instantiated variables and are of the form,

If condition was satisfy, then change tag A to tag B

If condition was satisfy, then change tag to tag Bs

where A and B are variables; the first transformation refers if a rule was satisfy a condition on word **(W)** with current tag (**B) i.e (W/A)**then the rule replaces current tag (**A**) with resulting tag (**B**). The second transformation refers that if the rule triggers on a word (W regardless of the current tag) then the rule tags the word with resulting tag (**B**). Therefore, a set of all acceptable rules are generated from all possible instantiations.

Finally rule based approach the rules can be considers as core aspect of the approach (Eric, 1995). Rules are derived either automatically or manually. Basically rule based approach used lexical and contextual information of the words to derive both the lexical and contextual rules during learning phase. According to Eric (1995) the main advantage of rule based approach was storage capacity reduction, easy to update and better portability from one set to another. However rule based tagger have some limitation like difficulty on constructing rules and the rules are not robust enough and could be time consuming.

2.3.2 Averaged Perceptron approach

The perceptron algorithm is one of the oldest algorithms in machine learning (Michael & Nigel, 2002). It is an incredibly simple algorithm to implement, and yet it has been shown to be competitive with more recent learning methods such as support vector machines (Michael & Nigel, 2002). The researcher describes the algorithm as an alternative to maximum-entropy models or conditional random fields (CRFs) for training tagging models. Yet, a brief theory justifying of algorithm with proof of convergence of perceptron algorithm for classification problems was provided. On the other hand Michael & Nigel (2002) explained how perceptron and voted perceptron algorithms can be used for parsing and tagging problems.

A range of syntactic processing tasks using a general statistical framework that consist a global linear model, trained by the generalized perceptron together with a generic beam search decoder was also implemented (Zhang & Clark, 2010). The researcher applies the framework to word segmentation, joint segmentation and POS tagging, dependency parsing, and phrase-structure parsing. The researcher further contended that the framework offers the freedom to define arbitrary features which can make alternative training and decoding algorithms.

Perceptron algorithms belong to a broad family of on-line learning algorithms and confess a large number of variants. The algorithm learns a linear separator by processing the training sample in an on-line fashion. The algorithm examines only a single element taken from an example. Thus, at each round, the current hypothesis is updated if it makes a mistake that is if it incorrectly classifies the new training point processed (Mathiaw, 2013).

Perceptron is a classic learning algorithm for neural model of learning, which is one of the learning algorithms that is incredibly simple and works very well in practice. It also uses features that behaves like an HMM tagger and the standard Viterbi decoding Collin (2002). Its main goal is to have a fast implementation for tagging large amounts of data. The features are modeled using feature method of the form feature/tag for a history hi and a tag ti, in the way pioneer by Mathiaw (2013). The history hi is a complex object modeling different aspects of the sequence being tagged. It may contain previously assigned tags in the sequence to be annotated, or following assigned tag in a sequence to be annotated and it can be also other contextual features like mor-

phological property of candidate token, context of previous and following words. Intuitively, the job of the training algorithm is to find out which feature functions are good indicators that a certain tag ti is associated with a certain history hi.

Average perceptron model consists of feature functions paired with feature weight which is estimated during training. As explained the main target of the approach is remain to find the best feature that are assumed to be good indicators that a certain tag t_i is associated with certain context representation h_i. Thus, the scoring method with average perceptron algorithm is defined over the entire sequences. The entire sequences in the concept of PoS tagging task means sentences. For a sequence of length n sentence in a model with d feature functions, the scoring function for the sentence w (w_1, w_2...wn) mathematically can be defined as follows:

$$score_w(t) = \sum_{i=1}^{n} w_i * x_i$$

Where w_i is the current weight and xi the input sequence of the feature functions. Accordingly the feature function is computed on the bases of a predefined feature template. For each candidate tokens a features template was applied with the concept of a history hi. After each and every features of candidate token was computed dot product of current weight and the input feature is preformed as in the above mathematical expression. In every iteration range the weight of the algorithm performs update function for the candidate tag sequence and the true tag sequence. Whenever the proposed tag was assumed to be incorrect tag then the proposed tag that leads the model to false prediction will be penalized and the true tag was rewarded (Mathiaw, 2013).

From the mathematical definition of scoring function the highest scoring sequence of tags can be computed or approximated as:

$$\tilde{t} = \arg max_t \ score_w(t)$$

Training the model is done in an error-driven fashion, tagging each sequence in the training data with the current model, and adding to feature weights, and then difference between corresponding feature function for correct tagging and the model's tagging (Mathiaw, 2013). The below

snippet part of the training algorithm shows perceptron training algorithm on training dataset sequence. All parameters of the model are initialized to weight = 0.

```
Method train (nr_iter, examples):
    for i in range of the iteration:
        for features, true_tag in examples:
        guess = self.predict(features)
    if guess != true_tag:
          for f in features:
              self.weights[f][true_tag]+=1
              self.weights[f][guess]-=1
   random.shuffle(examples)
```

As explained Average perceptron algorithm consider one example at a time instead of considering the entire data at once during training. Second it is error-driven, which means as long as it is doing very well, it doesn't bother updating its parameters. The algorithm maintains a "guess" at good parameters (weight and bias) as it runs. Then for a given example it makes a prediction. It checks to see if this prediction is correct or not. Whenever prediction is correct, the algorithm does nothing. If prediction is incorrect does it changes its parameters, and it changes them in such a way that it would do better on the example next time around. Once it hits the last example in the training set, it loops around for a specified number of iterations.

Recent research shows that averaged perceptron algorithm shows PoS tagging state-of-the-art accuracy for feature rich languages and morphological complex language with Stagger (Loftsson & Östling, 2013). On the other hand Matthew (2013) compares result of averaged perceptron tagger with the NLTK tagger and pattern tagger. As a result the averaged perceptron tagger was performing with the state of the art for PoS tagging problem.

In general averaged perceptron approach is a supervised learning approach. It uses feature rich sets of function to determine the best tags sequence for a candidate token. It uses weighted aver-

age score values to determine the best hit and also uses the frequency word/tag distribution to hold the most common word/tags sequence in the training corpus. For instance when we feed unseen dataset to trained averaged perceptron tagger first the candidate token was evaluated in the frequent wards look up table then the feature extracted automatically from the corpus content for the candidate token was examined. Averaged perceptron models have the advantage flexible features that can extend to intended language (Mathiaw, 2013).

2.3.3 Stochastic Approach

Stochastic approach is also called statistical approach which works by constructing a probabilistic model. Stochastic PoS taggers uses Markov model which captures lexical and contextual information (Nedjo & Liu, 2014). The two commonly used classes of statistical approach are Hidden Markov Model (HMM) (Gebregzabiher, 2010), (Mekuria, 2013) and Maximum Entropy Model (MaxEnt) (Nedjo & Liu, 2014). Both are models whose job is to assign word calss label or word class categories to corpora content in a sequence. The approaches use either simple or complex probability to resolve ambiguity on the bases of most likely interpretation. As like the other approaches stochastic approach can be trained using either supervised learning (Gebregzabiher, 2010), and (Mekuria, 2013) or unsupervised method. BeamWelch and Viterbe algorithms are the most commonly utilize algorithm in this approaches (Brants, 2000).

Stochastic taggers resolve tagging ambiguities using statistical property of the training corpus by computing probability of candidate token, having a given tag in a given context. For example, Unigram Tagger is a sequential tagger that uses simple statistical property to associate word class marker to candidate token with tag that is most likely to go with the candidate token in the training dataset. It uses a training corpus to decide which tag is most likely for each type. In particular, it assumes tag occurs most frequently with the candidate token is the most likely tag for. For example, if the training corpus contains the word "ᑎ ᐱ "(bela) as an Adjective (Aj) 5 times, and as a verb (V) 2 times, then it will assign the Adjective tag to any tokens whose type is "ᑎ ᐱ ". Unigram Tagger uses a ConditionalFreqDist() to record the most likely tag for each type. The train method constructs this conditional frequency distribution from a training corpus. Most of the time HMM based tagger preferred by different researchers and was proposed and implemented

for different languages including local languages like in Tigrigna (Gebregzabiher, 2010) and Kafi-noonoo (Mekuria, 2013).

Finally we can imitate the most common drawback of this approach as time constraint. The symbol probabilities are difficult to estimate for words. A substantial portion of text is composed of low frequency words. For these words, there are no enough observations make accurate estimates of symbol probabilities and words which are unknown when training have no observations at all. Another weakness of many stochastic taggers is their reliance upon hand-tagged corpora for training. While hand-tagged corpora do provide accurate estimates, they are very expensive to produce

2.3.4 Hybrid Approach

If we have several models that perform the same task, it might be reasonable to obtain better performance using combination of them than isolation (Asress, 2008; Gebregzabiher, 2010; Mekuria, 2013). By combine output of different models that form joint decisions regarding labeling of the input. The approach can be constructed either using combination of ANN and rule based approach (Asress, 2008) or stochastic and rule based approach (Gebregzabiher, 2010; Mekuria, 2013). The main intention of combining two or more models is to maximize performance of the model using the best features. In all case the reported performance was shown the hybrid model performance was higher than individual one. This is due to the fact that hybrid model considers the main drawback from one model and advantage of the other model. Thus, the main drawback of the first sequence will cover by the next one.

For this work purpose hybrid of averaged perceptron and rule based model was preferred. The main advantage of averaged perceptron was it uses two alternatives to assign tag for candidate token either using the most frequent word/tag in the training corpus or feature rich context. Its main drawback is, since it uses frequency of words in the first alternative it may assign tag incorrectly. Rule based approach uses lexical and contextual information about the candidate token to assign tag. Basically, it induces grammatical feature of the language during run time to deduce lexical and contextual rules that was expected to correct the incorrect assignment of tag during

initial state annotator. Another important thing in rule based tagger is it allows as combining models as initial state annotator to improve the tagger performance.

For this and other aspects of the two approaches a sequence of tagger combination was utilized for this work purposes. At the very beginning after unseen tokens are passed through the preprocessing stage the tokens are feed to averaged perceptron tagger as a sequence of tokens. Based on the aforementioned concept the averaged perceptron tagger assigns a tag to the candidate token. Since the tagger uses a weighted average to assign a tag for each and every candidate the last maximum averaged weight of the tag for the candidate token was returned. Then the final averaged weight was computed against the confidence level set in the averaged perceptron output analyzer. If the returned max weight was equals to the confidence level the tag was taken as a final tag otherwise a sequence of token was given to the rule based tagger for correction. Finally the output of the tagger was evaluated using confusion matrix against a reference corpus that was expected to be manually annotated.

2.4 Advantages of Part of Speech Tagging (PoS)

Beside the general and specific advantage of PoS tagging, large number of high level language processing tasks like parsing, spell checking and speech synthesis uses output of a tagger as preprocessing stage (Mekuria, 2013). The associated information to corpora contents is so important for providing significant information about a word and its neighbor. For instance with the help of PoS tag associated with the word 'ቦላ/V' we can pronounce as 'bela' (ቦላ)(hit her) and (ቦላ/Aj') beAl (tell her) if it is an adjective.

As long as we focus on part of speech tagging task it is helpful for to develop stemming algorithm in information retrieval because knowing word class categories help as which morphological affixes can take (Yonas, 2011). It is also helpful in expanding information retrieval (IR) applications. It also plays a significant role in word sense disambiguation algorithm (James & Daniel, 2006).

2.5 Part of Speech Tagging Model Evaluation Methods

In designing PoS tagger size of tag-set and sources knowledge added to corpora content takes significant portion whether the model is complex enough or simple. As reported in different researches most morphologically rich and complex languages require tag-set at low level for producing high performance.

Considering tagging strategies and state-of-the-art tagging performance accuracy for Tigrigna using rule based tagger, averaged perceptron tagger and hybrid approach several experiments were conducted. Following previous work, in all experiments we trained the model on different portion with small scale of the training dataset. Furthermore, following Gebregzabiher's (2010) work, considering morphological complexity and nature of the language we believe that Tigrigna requires tagger based on recommended course-grained tag-set. Course-grained Morpho-Syntactic tag-set of 22 tags appearing in the TGC (see chapter 2 Section 2.4).

The main evaluation mechanism we use for this work is accuracy measure (Asress, 2008), (Gebregzabiher, 2010) and (Mekuria, 2013). Confusion matrix was also used to compute performance of taggers output with reference corpus. Finally, a simple program was developed to compute in which candidate tokens the tagger most commonly miss-tagging.

2.6 Related Work

As explained earlier several PoS taggers were proposed locally and internationally. Among these Rule based, statistical and combined approaches are the most prominent one. The rule based approach uses a set of rule as fundamental building blocks of the tagger (Brill, 1995). The statistical taggers are easy to implement and require very less knowledge about the language. Neural network approach and averaged perceptron are also successfully implemented for different languages.

2.6.1 Previous Work on Foreign Languages

Achmid (1994) has already applied a two layer version of neural network approach to the problem of part-of-speech tagging. As the researcher call the tagger Net-Tagger consists of Multi-

layer perceptron and lexicon. The Net-Tagger was trained bases on supervised approach on 2 million words, and 100,000 separate words to testing on part of the penntree bank corpus.

To train the network the researcher uses modified backpropagation algorithm with momentum term. During the experiment networks with and without hidden layer was experimented. Network with hidden layer was more power full than network without hidden layer. Accordingly the network performance was reported 96.26% accuracy.

According to the researcher the input to the network contains all information like the preceding word (total of 3 preceding words 'p') and the following word (total of 2 words 'f'). Thus, for the word being tagged and the following word the lexicon part of speech probability$P(pos_j|word_i)$ is considered, however the lexicon part of speech probability doesn't have contextual influences. For the preceding word the activation value was used instead of the lexicon part of speech probability i.e. value of the output unit during the process was used instead of the lexical part of speech probability.

Each unit in the output layer corresponds to the tag in the tag set. Achmid reported that the network was learned during the training by adapting the weights of the connections between units to activate that output unit which represent the correct tag and deactivate all other output units. In the trained network the output unit with the highest activation indicates which tag should be attached to the currently processing word.

In both network (networks with and without hidden layer), single word tagging was performed by copying the tag probabilities of the current word and its neighbors into the input units, propagating the activations through the network to the output units and determining the output unit which had the highest activation. Then the tag corresponding to the unit was attached to current word. When training starts Due to the possibility of have incorrect output from the network, the network cannot feedback directly. Instead the actual output weighted average and the target output was used. At the beginning of the training, the weighting of the target output is high. However, it falls to zero during the training. Target activations are 0 for all output units, except unit that are corresponds to the correct tag, which is 1. If the differences between activation of output unit and target output is below 0.1 which is the threshold, error signal δ_{pj} was set to 0. The network

was forced to pay more attention to error signals with larger values. The tagging accuracy was increased by more than 1%.

According to the researcher experimental report score performance accuracy of Net-Tagger was compared to that of two other models (trigram tagger and HMM based tagger) in the same training and testing data set. Afterward, obtained experimental result was reported as the score performance accuracy of the Net-Tagger was 96.22%, score performance accuracy of the trigram was 96.6% and score performance accuracy of the HMM was 94.24%. Thus, the experimental report score performance accuracy of Net-Tagger shows that comparable performance to that of the trigram tagger and better than that of HMM tagger. The training time took two days to train the tagger in 10 different workstations and 12 minutes to test the tagger in the same machine. The time to train the network is considered as the main drawback of the tagger. Achmid further contended that due to much smaller number of parameters in the Net-Tagger the performance of the Net-Tagger was less affected by small amount of training data than that of trigram tagger.

2.6.2 Previous Work on Local Languages

Asress (2008) has conduct Part of Speech Tagger based on hybrid of (Neural Network and rule based) approach for Amharic language. The proposed tagger performs on the bases of two step process; first plain Amharic text passes through neural network and then rules was applied as error detector and corrector for ambiguity. According to the researcher BackPropagation Algorithm and Transformation based learning method was adopted for his work.

The text corpus used for his work was collected from Ethiopian Language Research center (ELRC), Addis Ababa as the main Sources of the text corpus. According to the researcher he was collected around 210,000 words from one ELRC project called 'The Annotation of Amharic News Document' which is meant to tag each Amharic word in its context with the most appropriate PoS manually. The text data used for this project was also collected from 'Walta Information Center'. The researcher uses more tag-set than the previous work proposed for Amharic language part of speech tagger; he uses 30 tag-set.

According to the experimental report the performance of the proposed tagger was evaluated based on the accuracy they assign the best tag for each word in the sentences. Three different

models were evaluated neural network, rule based and the hybrid one. According to the researchers experimental report each tagger performance accuracy was computed as 91% and 94% and 98%for rule-based, neural network and hybrid tagger, respectively.

The first attempt Part of speech tagger development for Tigrigna language was made 6 years ago by Gebregzabiher (2010) with hybrid model of (HMM and rule based) and experiment were con-ducted accordingly . The experimental have conducted using 10 fold cross validations. The rule based tagger incorporates different initial annotator and machine learning rules. The HMM model contains contextual and lexical models, and Viterbi algorithm is used to find the optimal part of speech tags for a sequence of words in the input sentence (X). The hybrid model was made as a sequence of HMM approaches followed by rule based to approach. The text corpus was domain specific (news sentences) contains 26,000 total words, 8000 words are distinct and 1000 sentences. A total of 36 Tag-sets was prepared and used to label Tigrigna text. To train and test the developed model supervised learning algorithm with translated Tigrigna letters was used.

Within the hybrid tagger HMM acts as an initial tagger assignment followed by rule based tagger as error detection and correction. Since the hybrid tagger works word level tag assignment window sized used as additional information on the Brill tagger is fixed to two when the labeled data fail the expression of fixed threshold values in the output analyzer. Three different approaches were adopted. Individually the taggers assign tag label to candidate token was made in sentence level. However, the hybrid model works tag assignment as word level.

The experiment has conducted with 25% for testing and 75% for training dataset. The performance of each tagger was measured up to 89.13% with HMM, 91.89%with Rule based and 95.88% to Hybrid tagger. Finally the researcher reported that the hybrid approach shows better performance score than the individual tagger alone.

As explained all tagger are trained and tested for small dataset from a single genre. Comparatively, as it is a first work in Tigrigna it is good attempt, however, both combined models are data driving models. Because, both models are data deriving approaches the model require more dataset to train and deduce their generalization to unlabeled dataset. In other term if the number of unknown words increases to the test dataset performance of the tagger is degrading. To start with

HMM tagger since it uses statistical models to compute best tag sequence of candidate tokens it requires large amount of data to train and to decide the best probability from the statistics collected within the training history. The Brill tagger is transformation-based error-driving approaches which automatically induce grammatical property of a language from training dataset which implies that it requires large amount of dataset to deduce permissible lexical and contextual rules and the dataset should be representative of different aspects of the language. However, the hybrid tagger was designed to assign word class category at word level which fails with the concept of sentence level tag assignment. However, considering the morphological complexity of Tigrigna, source of the corpus and in attempt to improve the tagger performance, it is good to have tagger that implement with morphological and contextual information of the language and tagger that uses the statistical history of the training corpus. However, all taggers are train and tested for a small dataset from a single genre that indicates the performance of the tagger was high but there is high level miss tag assignment.

The researcher combines default, unigram, bigram and trigram taggers as initial state annotator for Brill tagger. Basically the main ideal logic of combining tagger is to annotate unlabeled dataset and to create temporary corpus that helps the tagger to improve its performance. Thus, default tagger assigns a default tag for every candidate token even they are not in training corpus. Even if it is an obvious solution to increase performance of the tagger as a result it can increase possibility of wrong tag assignment to candidate tokes specifically if it integrates with statistical approaches. Unigram tagger uses simple statistical expression to assign the most likely tag to candidate token that are in training corpus. The most likely tag can also lead to wrong tag assignment since it counts only the most frequent word/tag assignment. The bigram tagger uses somewhat sophisticated statistical expression than unigram tagger; however, still it uses the statistical property to assign tag.

The designed model uses Latin translated scripts for Tigrigna fields to train and test the tagger and to prepare the annotated corpus. Thus, miss-translation of character can lead the tagger to unfair tag assignment and misconception. This can affect performance of the tagger. As the researchers stipulated the translated characters that have equivalent meaning with Tigrigna fiddles into the appendix most characters inhibit similar translation. Thus, the possibility of miss-

translation and miss tag assignment is high, and these can be taken as possible errors indicator within the tagger and possible indicator the tagger is not effective.

According to Hrafn and Robert (2013) state-of-the-art for tagging of morphologically rich languages is well below the 97+%. Thus, the obtained result was not bad. However, as mentioned because the taggers are designed from combination of data deriving models, it is the main drawback of the model. The researcher infers that, feeding dataset from source other than the trained genre degrades the tagger performance.

Mekuria (2013) has conduct part-of-speech tagger based on hybrid of (HMM and Rule based) for Kafi-noonoo language. The corpus was collected from two different datasets (from Kaffa Cultural and Tourism Bureau around 1000 proverbs was collected and 5 long reading passages taken from Grade 9 and 10 Kafi-noonoo student books). Thus, for training and testing purpose, 354 untagged Kafi-noonoo sentences are collected. To annotate the collected corpus an incremental annotated corpus preparation approach was used. For annotation Kafi-noonoo text corpora a total of 34 part-of-speech tag-set were used. The corpus was split in to the ratio of .9 and .1 to train and test the hybrid, HMM and rule-based taggers. i.e. the hybrid tagger is modeled as in the case of Gebregzabiher's work that means it combines the sequence of tagger as the HMM model as initial state annotator based on the specified threshold probability value and the rule-based tagger is used as error correction and detection mechanism. In both hybrid and individual tagger if unseen data was feed to the tagger it passes through preprocessing stage and assignment of most likely word class label to Kafi-noonoo texts was made at sentence level. Python and NLTK were used as developmental tools to set up the experiment. Three different approaches were proposed and designed.

The HMM method find the optimal part of speech tag sequence for a given word sequence, using a Viterbi algorithm using supervised learning mechanism. The process has three steps. First a tagged corpus as a training data was provided. Then sentence and word tokenizer was applied on the pre-tagged data to be tokenized both at sentence and word level respectively. Secondly, the statistical analyzer module computes both lexical and contextual probabilities from the pre-processed data and it stores them in the database. Finally, list of tags were stored in the database.

The lexical and contextual probability stored in the database provides information about the probability of each of the tag given a word and the contextual information about the word.

The TEL learning model was adopted for Kafi-noonoo language features such as prefix, the maximum length of character that can be deleted from the begging of the words to predict tag of unknown words from the existing one and the length of word that can be allowed before and after the given token to find the tag of the token based on contextual information.

Finally the researcher report performance accuracy of each model was reported 77.19%, 61.88% and 80.47% for Brill and HMM and hybrid tagger respectively.

Therefore, tagging is a process of assigning word class marker information to corpora contents that can provide further information about the word and yet has become a hot research area as it is significant is not trivial for language processing. Moreover, tagging is also important for other high level language processing task as an intermediate step such as parsing, semantics, translation, and many more. Many tagging research have been done and different approaches have been used for tagging, however most of them are conducted with Stochastic and rule based approaches for tackling the problem of tagging. Hence, there have been many researches done by combining the rule based and stochastic or ANN and rule based so that it would be possible to take the advantages from both approaches thereby improving the performance of the tagger.
Training and testing part of speech taggers with smaller dataset affects is performance, so it is more difficult to accept the result obtained as it may not reflect the reality (Gebregzabiher, 2010). However, training and testing PoS tagger with large enough dataset may show better representation of the reality and it is easy to accept result obtained by the tagger. Therefore, as far as the researcher's knowledge is concerned, there is no hybrid tagger work at sentence level for Tigrigna language so developing PoS tagger at sentence level for Tigrigna is importance considering the language morphological property and word ambiguity than of word level tagging.

CHAPTER THREE

LINGUISTIC PROPERTY OF TIGRIGNA LANGUAGE

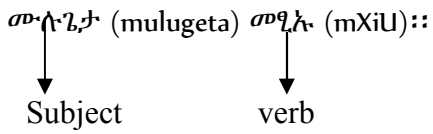
3.1 Overview

Tigrigna is a family group of Semitic branch of Afro-asiatic languages. Tigrigna is spoken by ethnic group of people in the north Ethiopia and central Eritrea. Speakers of the languages are called Tigrawot (feminine Tigrāweyti, male Tigraway, and plural Tegaru) (Tewelde, 2002).

Tigrigna has its own characterizing phonetics, and morphological properties. Even if Tigrigna uses Ethiopic written style and it uses similar letters to Amharic it has a set of letters that sound differently like [Oe](ዐ),[He] (ሐ),[Ke](ኸ),[a](ኣ) and [Qe](ቐ) which are not in Guragina or Amharic. Unlike Amharic that has many letters with the same pronunciation, for example ሀ (he), ሐ (He), are all pronounced as 'he', Tigrigna letters have unique pronunciation (Tewelde, 2002).

As explained in chapter 1 section 1 Tigrigna text document are written with a version of the Ge'ez script called ፊደል (fidel). These letters help Tigrigna users and Tigrigna speakers to construct meaningful ideas and share information in the form of sentences. Tigrigna sentences should have at least two components the subject and a finite verb. Tigrigna sentence structure follows subject (በዓል-ቤት) (beOel-Biet) object (ተስሓቢ) (tesHebi) verb (ግሲ) (gsi) word order (SOV) (Tsehaye, 1979), (Tewelde, 2002), (Teklu, 2008), (Tegay, 2014). The verb comes in the sentence final position and is the head of the verb phrase. For instances see the example

ሙሉጌታ መፂኡ። mulugeta mXiU። (mulugeta came.)



Tigrigna is morphologically richSemitic languages and it uses inflectional and derivational morphologies which results in very large numbers of variants for a single word (Yonas, 2011). As a result, a given root word can be found in several numbers of different forms. For instances, in the

2nd and 3rd person Tigrigna verb is inflected for gender as well as number. Eg. When the subject is you it can be a man (ንስኻ) (nsKa), women (ንስኺ) (nsKi), group of man (ንስኹም) (nsKum), and group of women (ንስክን) (nsKn). On the other hand the third person they (ንሳቶም/ንሳተን) (nsaten/nsatom) are applicable for both man and women. Table 3.1shows the inflectional properties of the root word (ስብር) (sbr) which means that 'break' in this case the root word was inflected for different forms like (gender, number, time) and will produce different meaning accordingly.

Table 3.1: Sample inflectional properties of words in Tigrigna language (Yonas, 2011)

Root	**Perfect form**	**Imperfect form**	**Gerund form**	**Imperative form**	**Causative form**	**Passive Form**
Sbr (ስብር) **(Broken)**	Sebere (ሰበረ) (he broke)	Tsebr (ተሰበረ) (broken)	Sebirka (ሰቢርካ) (You Broken)	Sber (ስበር) (break it)	Asbere (አስበረ)	Tesebere (ተሰበረ) (it was roken)

Derivational morphological properties of Tigrigna language describe how affixes are combining with word stems to derive new words (Yonas, 2011). Derivational affixes may affect the part-of-speech and meaning of a word. For instance,

A. The word mskr (ምስክር) is a noun so by adding the suffix '–net (ነት)' we will have the derived noun called mskrnet (ምስክርነት).
B. selam (ሰላም) in Tigrigna language ሰላም (selam) is a noun so let's look at how we create the derived noun from the word by adding the suffix '-awi (አዊ) to the noun called selam (ሰላም) we will have the derived noun called selamawi (ሰላማዊ).
C. hafti (ሃፍቲ) is also a noun so by adding the suffix 'am' (-አም)' to the word we will have the derived noun called 'haftam' (ሃፍታም)

3.2 Tigrigna Language Writing System

As we explained earlier Tigrigna documents are written with an adapted version of the Ge'ez script where originally developed for the Ge'ez language, during the 13rd century. Tigrigna written system is alpha-syllabic script in which each symbol represents a Consonant + Vowel syllable. The writing format of Ethiopic style is from left to right. i.e. the flow of ideas is from left to right (Tewelde, 2002).

Even if Tigrigna has its own written style and characters, it can be translated into its equivalent Latin representation by finding a Latin letter with similar sound in Tigrigna characters. This will helpful for the language to be readable and easier for people that can't read and understand Tigrigna language in general. However, this will have its own risk and constraint in the language. During translation minor error can cause misconceptions, whole or part from the document to be translated into a wrong manner and will lead the information receivers to have incorrect information. So it needs careful and series implementation and translation but it is not secure. For instance Tigrigna letters 'ሰ' have similar sound with Latin letter 'se' Thus; we can't assign the letter 'Se' with Latin letter sounds 'Se'. see Table 3.2 that will lead as to the representation of the seven order for letter 'ሰ' by combining the Latin letter's' with vowels as shown below.

Table 2.2: Sample of Tigrigna letter and their corresponding Latin characters (Omniglot, 2016)

Order	**1st**	**2nd**	**3rd**	**4th**	**5th**	**6th**	**7th**
Tigrigna letter	ሰ	ሱ	ሲ	ሳ	ሴ	ስ	ሶ
Equivalent Latin letter	se	su	si	sa	sie	s	So

Therefore, for the aforementioned reason we use Tigrigna letters (ፊደል) (fidel)without translation for this thesis work purposes.

Tigrigna text document consists of alphabets (ፊደል)(fidel), numbers (ቁፅሪ)(quxri) (numeric number and cardinal number), and punctuation marks (ስርዓተ-ነጥቢ) (srOete-neTbi). Through the next section, different part of Tigrigna text document was explained.

3.2.1 Tigrigna Alphabets (ፊደላት)(fidelat)

Tigrigna has its own alphabets each represent consonant-vowel syllables. They are sets of characters which are arranged in some sort of fashion used to construct words which are the fundamental building blocks of a sentence. Altogether Tigrigna has thirty seven (37) bases consonants with seven (7) vowels that change the basic phoneme of every consonant in to seven different character orders (Tsehaye, 1979). Some Tigrigna fidel (ፊደል) was present in Table 3.3 and the full Tigrigna language (ፊደል) was present in appendix C.

Table 3.3: Some Tigrigna language Fidels (ፊደል) Alphabets (Omniglot, 2016)

First ord		Second order		Third order		Fourth order		Fifth ord		Sixth or		Seventh order	
ሀ	he	ሁ	Hu	ሂ	hi	ሃ	ha	ሄ	hie	ህ	h	ሆ	ho
ለ	le	ሉ	Lu	ሊ	li	ላ	La	ሌ	lie	ል	l	ሎ	lo
ሐ	He	ሑ	Hu	ሒ	Hi	ሓ	Ha	ሔ	Hie	ሕ	H	ሖ	Ho
መ	me	መ	mu	ሚ	mi	ማ	ma	ሜ	mie	ም	m	ሞ	mo

3.2.2 Punctuation Marks (ሰርዓተ ነጥቢ)(srOete neTbi)

It is obvious that identification of punctuation mark in once language processing task is vital to know word and sentence demarcation. Tigrigna has its own punctuation marks which are used to demark boundaries of a sentence like (period (።)(አርባተ ነጥቢ)(arbOte nTbi)), boundaries of word from other word like (comma(፣)(neTela-serez)(Ethiopic Comma)) and so on. The Most commonly used Tigrigna language punctuation marks are;

1. Word separator (: or ፡) (ክልተ ነጥቢ) (klte neTbi) is used to separate one word from the other. Most of the time it was found in bible, however, currently it is rarely used. As, a result a single space is used to separate words instead of this punctuation marks.
2. Full stop (።) (አርባተ ነጥቢ) (arbaOte nTibi) is used to shows when an idea is finished.
3. Semicolon (፤) (ድርብ-ሰረዝ)(drb-serez)(Ethiopic semicolon) is used to connect two sentences in to one.
4. The list separator mark (፥) is used to list things, separate parts of a sentence, and indicate a pause in a sentence or question Like the other punctuation marks,

5. List mark indicator (፡-) is used at the beginning of the lists. The Meaning of each of the Punctuation marks are provides in the below table.

Table 3.4: Tigrigna Language Punctuations (Omniglot, 2016)

Name	symbols	Description
Preference colon	፡-	Beginning of the list mark
Quotation mark	“”	quote some words or sentences taken from other
Exclamation mark	!	End of an emphatic Declaration, or command.
Question mark	?	End of question
Semi colon	፤	Sentence connector
Full stop	።	End of sentence
Word separator	፡	Word separator
Ethiopic Question Mark	፧	Question Mark
Ethiopic Comma	፣	Comma

3.2.3 Number System

Tigrigna numbering system can be classified as ordinal numbers, cardinal numbers, fraction, and dual. Cardinal numbers are numbers used to compute arithmetic operation and for calendar purposes like ሓደ /Hade (one), ክልተ /klte (two) …ዓሰርተ /aserte (ten)…, ሽሕ/Shih (1000)…. On the other hand Tigrigna numbers are utilized as an adjective to express size noun. Ordinal numbers are numbers used to refer position of something for comparison purposes ቀዳማይ /qedamay (first) ካልኣይ /kalay (second) ሳልሳይ /salsay (third), ሙበልዓሰርተ ሓደ /mebel aserte Hade (eleventh) etc. Tigrigna fractions including ፍርቂ /frqi (half), ርብዒ/rbOi (quarter’) and ሲሶ /siso (one third)… which can be correspond to the English special numerals as attached with the words in bracket (Tsehaye, 1979).

Table 3.5: Partial parts of Tigrigna language numbering system

Tigrigna	Hade	klte	seleste	arbate	HamuSte	Shudushte	SewOete	Somonte	tSOete	Oeserte
	1	2	3	4	5	6	7	8	9	10

3.2.4 Tigrigna Sentence Structure

Sentence is a set of one or more words, grammatically linked to one another to express a statement, question, request, command or suggestion. It can be also thought as a group of words that in principle tell a complete thought which conveys enough meaning. Typically a sentence contains a subject and a predicate. In any language a sentences are constructed from a group of words arranged in a structured and non structured fashion (Tsehaye, 1979).

Every language has its own forms of sentence construction system and rules. Arrangement of words with sentences can be varied from language to language; even if the required information from the provided sentences was the same. As discussed the formal way of Tigrigna sentence word order is subject object verb (SOV). Let's see the examples that express our thought which is directly taken from Gebregzabiher's work for this example purpose only.

A. ሓውካ መፂኡ።/Hawka meTiu.

Your Brother came (has come).

B. ሓውካ ትማሊ ካብ ሽረ መፂኡ። /Hawka tmali kab Sre meTiu.

Your brother came (has come) from shire yesterday.

C. እቲ መፅሓፍ ኣብ ልዕሊ ዓራት ግበሮ።/Ati meTHaf ab lAli arat gbero.

Put the book on the bed

In the first two sentences **ሓውካ**/Hawka (your brother) functions as a subject, while the rest parts of the sentences are used as a predicate **መፂኡ**/meTiu (came (has come)) and **ትማሊካብሽረመፂኡ**/tmali kab xre me^Si^u (came (has come) from shire yesterday). The subject can be thought as a noun phrase which is used to mention some objects or persons and the predicate is used to say something true or false about the subject. These two elements of sentences are known as noun phrase (NP) and verb phrase (VP) which are headed as noun and verb respectively. In the third sentence, the subject is entrenched with the verb **ግበሮ**/gbero which is you 3rd person singular that acts as the noun phrase (NP) and the sentence '**እቲ መፅሓፍ ኣብ ልዕሊ ዓራት ግበሮ**/^ti me^SHaf ^ab l`li `arat gbero' acts as verb phrase (VP) which in turn is divided into noun phrase and verb phrase.

Structural way of constructing sentences will help language users to share ideas in simple and easy manner. As other languages constitute structured system of communication behavior Tigrigna also inhabits structured system of communication. Tigrigna has three major types of sentences namely, simple, complex and compound sentences (Tsehaye, 1979).

Simple Sentences: is a sentence type which has subject and one verb. Simple sentences can be further sub divided into verbal and copulative sentences. The verbal simple sentence is a sentence in which it has one subject and one verb only and the subject as well as the verb can be with or without modifier (Tsehaye, 1979).

e.g.

A. mulugeta meTiu (ሙሉጌታ መፂኡ::) (Mulugeta came.)

B.ms mulugeta kmeTA Ayu::(ምስ ሙሉጌታ ክመፅእ እዩ::)He will come with mulugeta.

In example 1 the sentence has one subject and one verb

mulugeta meTiu::

Subject (በ-ል ቤት) (noun) verb (ግስ)

Tesfay kab asmera kmTA Ayu. (ተስፋይካብአስመራክመፅእእዩ::) (Tesfay will come from Asmara.)

. ተስፋይ (tesfay) ካብ (kab) አስመራ(asmera) ክመፅእ(kmeTAእዩ (Ayu) ::( )

Subject (noun) (preposition) (noun) (Verb) (verb) punctuation mark

The example has one subject which is tesfay(ተስፋይ) and one verb which is Come(ክመፅእ)(kmeTA)

Compound Sentences: is a sentence that has two or more than two independent clauses which has equal grammatical statuses and conjoined the clauses using connectors (Tsehaye, 1979). The connectors can be associative, alternative or constructive connectors. For more detail information let's see the following examples.

A. mulugeta ymehar alo kemu wn Haftu tserH ala:: (ሙሉጌታ ይመሀር አሎ ከምኡ ውን ሐፍቱ ትሰርሕ ኣላ።)mulugeta is studying and his sister is working.

The above example contains two complete sentence connected with and **(ከምኡ ውን)** (kemu wn) connector. The connector shows association among the two sentences. While mulugeta is studying at the same time his sister is working.

(Mulugeta he-studies exist-he sister-his she-works exist-she)

B. Haden blaA haden gdefo:: (ተደሊካ ቢላ◦ ተደሊካ ግደፎ።) (Either you eat or leave it)

Here in the above example the two sentences are connected with alternative connectors that indicate and command the subject to do an alternative provided to it. i.e either the subject should it or should leave it.

C. genzeb aloni gn bzuH aykonen. (ገንዘብ አሎኒ ግን ብዙሕ ኣይኮነን።) (I have some money but it is not much.)

(Money exist-for-me but much not-enough)

D. Sewit Tbqti Aya mulugeta gn Himaq Ayu:: (ሸዊት ፅበቅቲ እያ ሙሉጌታ ግን ክፉእ እዩ።)

Shewit is pretty but mulugeta is ugly.

Complex sentences: is a sentence that contains two or more clauses, thus, at a least one clause is made dependent by subordination.

Latin translation sentence	Tigrigna sentence	English translation sentences
A. genzeb serika teasira.	ገንዘብ ሰሪቃ ተኣሲራ።	(After she was stole some money she was arrested.)

From the above two sentences we see time series consequences in which after she stole a money she was arrested for her action. The first sentences refer to actions performed by her (she stole some money) and the next sentences were indicating result of the action (she was arrested).

Latin translation sentence	Tigrigna sentence	English translation sentences
B. AtiTel zketele bayteasiru.	እቲጠልዝቀተለሰብኣይኣሲሩ።	The man who killed a goat is jailed.

The above sentence contains two verbs: **ዝቀተለ**/zqetele (killed) and **ተኣሲሩ**/teasiru (jailed). In other word the sentence contains the main clause and subordinate clause. A subordinate clause is a clause which is dependent to the main clause (Tsehaye, 1979). In this case the main clause infers to the man was jailed (**እቲሰብኣይተኣሲሩ**/Ati sebay teasiru) and the subordinate clause is who stole goat (**ጤልዝሰረቐ**/Tel zsereqe).

3.2.5 Tigrigna Word Class Categories

Word class category or simply lexical category refers to classes in which a given word can be resided. The term word class is used to identify word behavior in the sentence or corpus. Tigrigna word class categories are classified to two major classes categories as 'closed' word class and 'open' word class (Tewelde, 2002) and (Teklu, 2008). Open class is classes in which a new words can be added to the class as the need arises. Tigrigna open word classes include noun, verb and adjectives (i.e. nouns, for instance, is infinite, since it is continually being expanded as new scientific discoveries are made, new products are developed, and new ideas are explored). However, Tigrigna closed word classes include only word with limited members. According to Tewelde (2002), Teklu (2008) Tigrigna preposition and conjunction is closed class.

For any language , to say that a word belongs to a given class or word is belong to the other class should have at least the basic criteria or simply rules in which how you classify the word in to their respective class (Tewelde , 2002; Teklu , 2008) . According to Sahilu (1998), Tewelde (2002), and Teklu (2008), the basic criteria that identifying word class category can be either based on syntactic forms of the word or forms of the word. In Tigrigna different linguists put word class category on the bases of their own belief and reasons. Thus, according to Sahilu (1998) work in the Tigrigna grammar book published in 1998, basic word class category was classified in to 'eight 8 ' categories. These are presented in table 3.6 as follows:

Table 3.6: Word Class Category in Tigrigna Language (Sahilu, 1998)

R/No	Word class	Example
1	Verb(ግስ)	Driv (ዘወረ), grow(ዓበየ), sing(ደረፈ)
2	Noun(ስም)	Sister (ሐፍተይ)፡Bus (ኣውቶቡስ)፣ house (ገዛ),
3	Adjective(ቅፅል)	Big (ዓቢ)፤ happy(ሕጉሰ) cleaver(ነበዝ)
4	Adverb(ተውሳከግስ)	Happily (ተሐጒሰ), recently (ቀረባ), soon
5	Preposition(መስተዋድድ)	of, over (ልዕሊ), with (ምስ), in (ትሕቲ)
6	Pronoun(ከንድ ስም)	He (ንሱ), she(ንሳ)
7	Conjunction ()	And (ከምኡ ውን), or (ወይ ድማ)
8	Interjection ()	Wow (ዋው), gosh (ነሽ)

In contrast to Sahilu (1998) in the modern Tigrigna language grammar book by Teklu (2008) states that word class category was classified in to five (5) see Table 3.6. Teklu (2008) integrates some of the word class categories are merged into one class. For instance noun and pronoun are merged to one class called noun, and Preposition cannot be stand by itself as one class. Through-out, this section we will follow the major word class stated in the work of Teklu (2010) and we will try to see them accordingly. The explanation for conjunction, interjection and determiner was taken from the work of Sahilu (19989).

Table 3.7: Part Of Speech Class InTigrigna Language (Teklu, 2008)

RN	**Word Class**	**Example**
1	Verb(ግስ)	Sing (ደረፈ), think (ሐሰበ), break (ሰበረ), go(ከደ)
2	Noun(ስም)	Mountain (ነቦ), Kalayou (ካላዩ)
3	Adjective(ቅፅል)	Foolish (ሕሙም), talented (ነበዝ)
4	Adverb(ተውሳከ ግስ)	Soon (ቀረባ), then (ድሒር)
5	Preposition(መስተዋድድ)	At (ኣብ)፤with (ምስ)

Noun (ስም) ፡ as like the other language Tigrigna nouns are name given for people, places, or other thing. Tigrigna noun can function as subject or object of transitive verb or complement of preposition (Tewelde, 2002).

e.g. ሙሉጌታ (mulugeta)፣ ገመል(gemel)(camel)፣ (ዓዲግራት) (adigrat) ፣ ድፍረት(dfret)፣ ድቃስ(dqas).. Noun can includes concrete terms like car and goat, abstract like relationship. Noun can be also sub divided in to the following sub class.

Common noun (ነይ ሐባር ስም) (nay Habar sm**)፡** is a noun which is used tocall a group of things that share common properties however it is not used to call specific things.

For example, People (ሰብ)(seb), food (ምግቢ) (mgbi), birds (ዐፍ)(Ouf)

Concrete Noun (ግዙፍ ስም) (gzuf sm) and **Abstract Noun**(ረቂቅ ስም) (reqiq sm): is a sub class of the major class noun. Both are used to represent things that can be touched and seen (concrete noun) as well as things that are abstract which can't touch and seen (Abstract noun). The example explores more about abstract and concrete nouns.

E.g. abstract nouns

ቁልነት(qulOnet) (childish) እንስሳነት(Anssanet)(animalism)ሰብአዊነት (sebawinet)(humanity)

ሙሉጌታ ገና ቁልዕነቱ ኣይወድአን፡፡ (mulugeta gena qulOnetu aywedan.) (mulugeta acts like a child.)

E.g. Concrete noun

Lion (አንበሳ)(anbesa), people (ሰብ)(seb), infant (ቆልዓ)(qolOa)

ትማሊ አብ ከተማ ዓዲ-ግራት ኣንበሳቆልዓነኪሱ። (tmali ab ketema Oedi-grat anbesa qolOe nekisu።) yesterday in Adigrat city a kid was killed by lion. Here the lion (ኣንበሳ)(anbesa), and the ckid (**ቆልዓ**)(qolOa)are nouns.

Noun pluralization (ረባሕታ ስም) (rebHta sm):

Tigrigna noun has plural forms to represent a number of things that share common characteristics. However, in Tigrigna as explained in the work of Teklu (2008) the most complex and difficult part of the language is there is no common system of converting a singular form to its plural forms. Thus, Tigrigna language noun can be converted in to its plural form of three different

ways like (gender, number and action). In the next example some Tigrigna language nouns pluralization form was present,

Seas (ባሕር-ታት)(baHrtat) here by adding the letters (ታት)(tat) to the singular form of sea (ባሕሪ)(baHri) we create a the plural form seas (ባሕር-ታት)(baHrtat)

Teachers (መምህራን)(memhran)here by adding the letters (-ኣን)(an)to a singular form of teacher (መምህር)(memhr) we create a pluralform Teachers (መምህራን)(memhran)

Stones(ኣእማን)(aAman)here by adding the letters (-ኣን)(an) to the singular form of stone (እምኒ) (Amni)we create a plural form stones **(ኣእማን**)(aAman)

Photos (ስእልታት)(sAltat)here by adding the letters (-ታት)(tat)to the singular form of photo (ስእሊ)(sAli)we create a plural form of photos (ስእል-ታት) (sAltat)

In general Tigrigna language nouns are inflected for number, definite and genders. Mostly noun pluralization is found by adding suffix ‘at’ (ኣት) or ‘yan’(ያን). Commonly affixes like ‘tat’(ታት), ‘at’(ኣት), an’ (ኣን)‘ot’(ኦት), ‘ti’(ቲ) and ‘wti’(ውቲ). Grammatically, Tigrigna language gender is belongs to either feminine or masculine, thus, all Tigrigna language gender are expressed in either of them.

Pronoun (ክንድ ስም) (knd sm)፡most of the time Tigrigna pronoun are used to replace noun in.For instance, see the below example

e.g. በላይኣንበሳቀቲሉ:: (belay anbesa qetilu) Belay kill a lion.

In this case belay (በላይ)(belay) is a noun and the noun was replaced in the next example by its equivalent pronoun.

ንሱኣንበሳቀቲሉ፡፡ (**nsu** anbesa qetilu)**He** kill a lion. In this sentence the subject of the sentence was **ንሱ**(nsu) ’he’ and it represent belay (በላይ) in the above example. So, in the above example it takes the direct name and in this case it takes the form of pronoun.

Adjective (ቅፅል) (qXl): adjectives are one of the four major word classes, and its main purpose is to give clear explanation for a noun and determine a noun as well (i.e they give us more information about people, animals or things represented by nouns and pronoun). Most Tigrigna adjective was found in front of a noun. On the other hand adjective modifies nouns. For instance, ቀይሕ (qeyH) (red), ፀሊም (Xelim) (black), ንእሽተይ (nAStey) (small), ዓብይ (Oeby) (large), etc are adjectives. As the other Tigrigna word pluralization form, Tigrigna adjective also shares the behavior of pluralization to express size, gender, and other forms. As stated the most common way of creating Tigrigna adjective plural form is by adding the letter '-ቲ ' in to the word . For instance in the word photographer (ሰአሊ) (sali) by adding the letter '-ቲ ' we have photographers (ሰኣልቲ)(salti) , for the word killer (ቀታሊ)(qetali) we have the plural word killers (ቀተልቲ) . In addition to the aforementioned way of pluralization mechanism there are also different ways to create Tigrigna adjectives. One way of making plural adjective is adding affixes (-አ፣-ኣት፤-ኦት፤) to a given word. A detail explanation was given in Table 3.8 how the word ሽቃሊ (Seqali) labor can be change its form using some of the aforementioned affixes.

Table 3.8: Sample Tigrigna Adjective Words Pluralization Technique (Teklu, 2008)

	-አ		-ኣት		-ኦት
singular	plural	singular	plural	singular	Plural
ሽቃሊ (labor-	ሽቃሎ	ክቡር	ክቡራት	ዘበናይ (fashion)	ዘበነቶት

Table 3.8 shows how Tigrigna adjectives can take different form to create their plural form. On the other hand see the next example that gives us detail explanation of the above forms of adjective on how to modify or explain nouns. ፅቡቅወዲ። (Xbuq wedi):: handsome boy . Here the word (ፅቡቅ)(Xbuq) handsome will explain to us the main characteristics of the noun ወዲ (boy)(wedi) in which the boy is handsome . ፅብቅቲጋል። (Xbqti gal) pretty girl; Here is also sharing the same form as in the above sentence explanation. The word ፅብቅቲ (Xbqti) which is an adjective gives as the girl is pretty. ሓፃርጋል።(HaXar gal) (Short girl.). The girl is short so here the word ሓፃር (HaXar) (short) is used as adjective to explain the girl height look.

Table 3: Sub Class of Adjective in Tigrigna Language (Teklu, 2008)

sub class (ክፍልታትቅፅል)	Description (ግልጋሎት)	Example
ቅርፂ(qrXi)(form)	Show the shape of something	ከቢብ፣ፀፊሕ፣ሞልማላ…
ሕብሪ(Hbri)(color)	Shows the color of some thing	ቀይሕ፣ፀሊም…
ኩነት(kunet)(action)	Show the manner of something	ሐያል፣ድኹም፣ዝሑል…
መጠን(mTen) (size)	Describes the volume of something	ሐፂር፣ነዊሕ…
በህሪ(bahri)(manner)	Describes behavior of something	ጨካን፣ለዋህ…
በዝሒ(bzHi)(number)	Show the number of something	ሐደ፣ክልተ፣…
ደረጃ(dereja)(Stage)	Describe something in comparison to others	ቀዳማይ፡ካልኣይ፣…

Verb (ግሲ) (gsi): Tigrigna sentence has subject, object, verb (SOV) word order to construct Tigrigna sentence. Mostly word comes at the last of a sequence is a verb. Thus, Tigrigna verb is found at the last a sentence. Verbs of Tigrigna have subject and object pronominal affixes attached to them (Tewelde, 2002). Tigrigna verbs have two types of ending: one relating to the subject and one relating to the object. Thus the affix attached to the verb can simultaneously agree with the subject or the object. The agreement can illustrate with person, number, and gender. Tigrigna verb describes subject's action or state within a sentence. Thus, without verb any sentences can't ends its concept and can't translate complete information. Tigrigna verb has three forms namely: transitive verb, intransitive verb and intensive verb (Tewelde, 2002). The examples were taken from the work of Teklu (2008) to elaborator how Tigrigna verb was used in a sentence.

A. አቲከልቢሞይቱ።።(Ati kelbi moytu።) the dog was dying. Here the word dying (ሞይቱ (moytu))was found at the end of the sentences and it refers to the action were the dog was dying.
B. ሃይለአንበሳቀቲሉ።። (hayle anbesa qetilu።:) Hayle kills a lion. Here is also the last word is ቀ ቲሉ and it refers to the action in which hayle was done in the past.
C. ሙሉጌታ ክዳን ሐፂቡ።። (mulugeta kdan HeXibu።) Mulugeta wash cloth. The same is true as in the above case the last word (ሐፂቡ) in the words sequence is also a verb which express the action which was done by subject mulugeta.

ተሳጋሪ ግሲ(tesagari gsi) (Transitive verbs): this kind of verb indicates that the action which was done by one subject can be transfer to the action receivers. For instance

> **ሙሉጌታ ጥርሙስ ሰቢሩ።።** (mulugieta Trmus sebiru::) Mulugeta breaks the bottle. In this example break (**ሰቢሩ**)(sebiru) is the verb and it is also indicate an action. As stated the verb shows as the action which is break which is directly apply to the bottle (the battle was broken). On the other hand the verb indicates that the action was done by mulugeta. (The bottle was broken by mulugeta). Table3.10 shows the detail explanation on how the action was passed.

Table 3.10: Transitive Verbs

Action doer (ፈፃሚ ግብሪ) subject (ባ-ል ቤት)	Action (ግብሪ)(gbri)	Action receiver (ተስሐቢ)(tesHabi)
ሙሉጌታ (mulugieta)	ሰቢሩ(sebiru)	ጥርሙስ (Trmus)

ዘይተሳጋሪ ግሲ (zeytesagari gsi) (Non transitive verbs): are kinds of verb which inhabit the reveres properties of transitive verbs. Here the action was not transferred to other. For instances

> **ሙሉጌታ ሰኒፉ።።**(mulugieta senifu::) Mulugeta was tiered. In this case the action was only re-fers to the action doer and can't be transferred to others.

In general since Tigrigna verb incorporate different behavior we can simply indentified it from a given sentences. The main characteristics of Tigrigna verb is stated below

1. Most verbs are found at the end of sentences.
2. In a given sentence verbs indicates an action done by subject of a sentences and also it is always agree with the doer of the action. For instance in example 'c' the verb is washes (ሐፂቡ) and the attached suffix indicated number of the subject doer. In simple manner the doer of the action is 'ሐፂቡ-**ኡ**'. As indicated in bold letter the attached vowel stipulated at the end of the word inform as action doer is a feminine and it is also singular in number.

Adverb (ተውሳከግሲ) (tewsake gsi): adverbs are words used to express and modify verbs, adjectives, sentences, clauses and other adverbs (Tewelde, 2002). It also qualifies a verb, adjective and other adverbs too. In Tigrigna language all verb modifiers or verb phrase are usually express time, location, manner, and number. As mentioned Tigrigna adverbs are limited to number, thus, adverbs are classified under closed word categories. i.e we can't create other words with them and we

can't create adverbs from other words too. In addition to that we can't pluralize adverbs in time gender and numbers. Here are some examples of adverbs taken from the work of (Teklu, 2008).

Table 3.11: Sample Example of Tigrigna Adverb (Teklu, 2008)

እውን(Awn) (also)	ንስክላ (nskla) (Escape)
ገና (gena) (Yet)	እሺ (ASi) (OK)
እምቢ (Ambi) (No)	ንየው (nyew) (go away)
ሕራይ (Hray) (OK)	መሊሱ (melisu) (Respond)

In general Tigrigna adverbs are used to modify Tigrigna verbs. We prefer to uses sentence to elastrator how Tigrigna adverbs are used in Tigrigna sentence. The next example provides us detail and clear explanation to the question how.

How something happens or is done (**Action**)

Ati bered nab may teqeyru::አቲ በረድ ናብ ማይ ተቀይሩ:: the ice was changed in to water.

Where something happens (**place**):

I live**here (አብዚ** እየ ዝነብር::)(abzi Aye zinebr::).

When or how often something happens (**time** and **frequency**):

They visited us**yesterday**. (**ትማሊ** እዮም ርእዮምና::) (tmali Ayom rAyomna::)

Preposition(መስተዋድድ) (mestewadd)**:**Tigrigna preposition refers to the closed set of words which precede nominal or nominal phrases and can express relationship or association among things, person or events etc or others(Tewelde, 2002)such as *after (ድሕሪ)(dhri), in(ውሽጢ)(wSTi), to(ከሳብ)(ksaO), on (ልዕሊ)(lOli),* and *with(ምስ)(ms).* Most of the time they are used in front of nouns or pronouns and they show the relationship between the noun or pronoun and other words in a sentence. For example, the position of something, the time when something happens, or the way in which something is done. Prepositions are most commonly followed by a noun phrase or pronoun.

Tigrigna preposition are limited in number. i.e we can't create new preposition from other words and we can't create other preposition from preposition itself (Teklu, 2008). In addition to that, we can't also pluralize them in terms of sex, number, manner and time. In Tigrigna preposition doesn't constitute its own meaning. Preposition further subdivided in to two sub class Table 3.12 explore the detail explanation and example of Tigrigna preposition.

Table 3.12: Sample Examples Of Tigrigna Preposition (Teklu, 2008)

Preposition that has two letters	Preposition that has more than two letters
ካብ (from)	ውሽጢ(in)
ምስ(with)	ቅድሚ(before)
ከምዝ(like)	ልዕሊ(on)
ናብ(to)	ከባቢ(around)
ስለ(due)	ክንዲ(instead)

The next two examples give as detail explanation on how preposition utilized in Tigrigna sentences. Sometimes preposition are used to indicate position of something. In the example the word **ትሕቲ (tHti)** is preposition, which is used to determine that the bag is under the chair. Basically, the word **ትሕቲ (tHti)** indicates that the position of the bag.

- Her bag was under the chair. (ቦርስኣኣብቲ**ትሕቲ**ጠረጴዛኣሎ።።)(borsa zabti tHti TerePeza alo::)

On the other hand Tigrigna preposition are used to express time in which something is happen. This can be clearly expressed as in the next example. If we closely look at the sentence, its main concept is arriving and the answer to when did they arrive is provided by the word shaded in bold which is attached in front of the word Sunday. The preposition attached to the noun Sunday give as detail information about the subject which was arrive on Sunday.

- They arrived on Sunday. (**ብ**ሰንበትበ**ጺ**ሖም።።) (bsenbet beXiHom::)

Conjunction (መስተፃምር)(mesteXamr): Tigrigna conjunctions are also called connective words. Most conjunction uses as an association among two or more Tigrigna sentences and word. Basically, their main purpose is to show how two things are associate together. Some examples of Tigrigna language conjunctions are and (kemu-wn), *because (mknyatu), but (koynu-gn)*ወይ-ድማ(wey-dma)(or). Tigrigna language conjunctions are used to connect phrases, clauses,

and sentences. The following example expresses the relationship among the two phrases. The relationship among the two phrases is brotherhood so kalayou's brother is mulugeta. The same is true the reveres.

Mulugeta and Kalayou are brothers. (ሙሉጌታ-ን ካላዩን ኣሕዋት እዮም።።)(mulugietan kalayun aHwat Ayom::)

Interjection: are words used to express feeling or emotion; mostly exclamation words (e.g. goS!(ጎሽ) wow!(ዎው) ohoy!(ኦሆይ)), which show people's reactions to events and situations:

ዓሸ! ንፉዕ ዝወደይ(OeS! nfuO zwedey) (wow! you (my son) excellent!) Summary of Tigrigna language major word class on the bases of their respective class for the closed and open word class label was presented in Table 3.13 and 3.14 respectively.

Table 3.13: Tigrigna Closed Word Class Summary (Tewelde, 2002)

Closed Word Classes				
Determiner	Pronoun	Preposition	Conjunction	Adverb
እቲ፣ እዚ፡	ንሱ፡ንሳ፡	ትሕሪ፡	ከምኡውን፡	ሕጂ፡
ኣበቲ፡ ኣብኡ	ነሳቶም፡ንሳተን	ልዕሊ፡ ኣብ፡ ካብ	ስለ-ዝኮነውን፡	ድሓር፡

Table 3.14: Tigrigna Open Word Class Summary (Tewelde, 2002)

Noun	Verb	Adjective
Abstract: ፍረሒ፡ጨዋተ	Transitive: ነኪሱ፡ ሰሪቁ	Descriptive: ሰነፍ፡ ህኩይ
Concrete: ጠረጴዛ፡ ነቦ	Intransitive: በከየ፡ኮርዩ	Comparative: ዝሓሸ፡
Common: ሰብ፡ገረብ	Modal: ክኢሉ	Superlative: ዝነውሓ፡ ዝሐፀረ
Proper: ሙሉጌታ፡ ካላዩ	Auxiliary: እዩ፡እዮም፡ ኣሎ	

3.3 Tigrigna Language Text Processing Challenges

As discussed some properties of Tigrigna language in general and word class in particular it is a little bit difficult to utilize Tigrigna language text document in the language technology. This can be the main limitation and requires language expert interference in the language processing task. On the other hand there is also limitation of resources. Basically, when we come to the work of

part of speech task, due to the morphological complexity and nature of the language the following limitation and challenge was expected. One main challenge can be, there is no any mechanism to distinction between upper and lower case letters. For this reason, the complexity of the language is high; in terms of word identification and identification of word class categories, unless we have word feature information like (lexical categories of the words) or other information attached to the words or completely in the corpus. Some of the main challenges are explained as follow:

Characters Redundancy: different letter are used to represent similar sound (Hafte, 2009) and show similar usage in Tigrigna language text documents. E.g. letters ("ሀ" and "ኀ"); ("ጸ" and "ፀ"); ("ሰ" and "ሠ") have similar usage in Tigrigna language text document. For instance the word ፀዓረ(XeOare) and ጸዓረ (xeOare) provides the same significance in a give Tigrigna language sentence. But the letter is different. The same is true as in the case of the above explanation ሰበረ(sebere) and ሠበረ(sebere)shares the same meaning but letters are different.

Spelling variation of the same word: Tigrigna word can be spelled differently using different spelling variation. For instance, when the word "Television" is can be written as "ተለቢዝን"(telbiZn), "ተለቭዥን (televZn)or "ተለብጅን"(teleBjn) in different Tigrigna language text document.All these words areused to mean the word "television" in Tigrigna this will increase the complexity of the language.

Abbreviation: Tigrigna abbreviation follows different format; some time full stop'.' is used for abbreviation, other time backslash '/' is used and sometimes abbreviation can be written without separators (ዓም). For example, (ዓመተምህረት) can be written as "ዓ/ም"or "ዓ-ም" or ዓ.ም. The inconsistency in the abbreviation format creates discrepancy in text processes and text normalization.

Compound words: Tigrigna compound words are written using hyphens as connector. For example, "ቤት-ትምህርቲ" school, "ቤተ-መፃሕፍቲ" (library), "ቤት-ትምህርት" (school), and ኣብያተ-ፅሕፈት (office) are compound words. So this can create problems in text processing and text normalization when they come without hyphen.

3.4 Tag, Tag-setand Tagging

Tag-set refers to a collection of tags which is used to mark word classes of a text corpus. Morpho-Syntactic tag-set is tag-set which inhabits morphological and syntactic information together about the text document collection (Tsegay, 2014).

Tag is a label or simply a marker which is attached to a given token to show part of speech classes as a word with a given text document corpus. According to (Tsegay, 2014) there are different kinds of tags. These are Part of speech tags, phonetic tags, and semantic tags and so on. Tag or simply class markers can be attached to corpora content either manually or automatically (Gebregzabiher, 2010), (Mekuria, 2013). In addition to that, as discussed in chapter 3 section 3.2 word class category markers can be attached to corpora content either in word label or in sentence label as. Thus, the process of attaching word class markers for corpora content is called TAGGING. There are two well prepared tag-sets available for Tigrigna to automatic text annotation purpose. These are tag-set prepared by Gebregzabiher (2010) utilized in the first Tigrigna Tagger developed in Addis Abeba University and the Morpho-Syntactic tag-set prepared by Tsgay (2014) called TIGTAGS. TIGTAGS Tag-sets are prepared for Tigrigna texts document to annotate automatically at three different levels. These are Tag-set at Course-Grained Level, Fine-Grained and Very-Fine-Grained Morpho-Syntactic tag-set. As stated earlier Morpho-Syntactic tag-set are tag-sets incorporate morphological and syntactic information together. In more general term the Morpho-Syntactic tag-set allows us to use both advantages of morphological property and syntactic property of the candidate tokens.

Tag-set size was the main determinant of the utilization of the tag-set. Moreover, the size of the tag-set can lead us to more general result and more specific result. According to Tsegay (2014) size of the three basic Tag-set categories is 18 tag-sets were course-grained Morpho-Syntactic tag-set, 105 fine-grained Morpho-Syntactic tag-set at high level and one is a very fine-grained tag-set of 139 Morpho-Syntactic Tag-set at low level. Gebregzabiher (2010) uses 10 basic tags and 26 sub tags collectively the researcher prepares 36 tags.

Up to the researchers knowledge tag label should be simple, easy and not complex (i.e if the representation label is so complexity and includes too much character it may be confusing). Due to

the morphological complexity and nature of Tigrigna , it is better and convenient to use tag set at course grained to start and understand the language property that are silent for language processing task . The same is true using tag-set at fine-grained level require Tigrigna language detail knowledge and time to encode all possible knowledge's. Using Tag-set at course-grained level helps us to minimize corpus preparation time, linguistic expert effort, and cost as compared to that of the other two tag-set. On the other hand, if we consider some property of Tigrigna we shouldn't leave the other property too. So, as far as the researcher knowledge concerns, and the language complexity and resources limitation of this research works purpose we adopt tag-set from the work of (Tsegay, 2014) at course-grained level. The tag-set includes 22 course grained labels including (Abbreviation <Ab>, Residual <Rs>, Symbols <Sy>, Foreign words <Fn> compound <Cp>) and so on. Table 3.15 contains detail explanation of each label alongside with their example.

Table 3.15: Tigrigna Language Tags (Tsegay, 2014)

RN	Word class	Tags	Example
1	Noun	N	mulugieta(ሙሉጌታ),ketema(ከተማ), africa(አፍሪካ)
2	Pronoun	Pn	nsu(ንሱ) (He), nsa(ንሳ) (she)
3	Verb	V	beXiHu(በፂሑ)(Arrive):qetilu(ቀቲሉ) (He killed),
4	Adjective	Aj	Xelim(ፀሊም)(Blak),qeyiH(ቀይሕ) (Red)
5	Adverb	Av	dHri (ድሕሪ) (After), wiSXi(ውሽጢ) (in)
6	Preposition	Pp	bkaliA(ብካሊኡ) (on the other hand)
7	Conjunction	Cj	kalayunkokebn(ካላዩንኮኩብን)(kalayouand kokeb)
8	Determiner	Dt	Ati(እቲ) (That), Atiom(እቲኦም) (these),
9	Interjection	Ij	OeS zwedey(ዓሸ! ዝወደይ) (Bravo! my son)
10	Punctuation	Pu	:-, :, ::, ?, !
11	Numerals	Nu	1(ሓደ),2(ክልተ),3(ሰለስተ)…qedemay(ቀዳማይ)(first)
12	Phrasal noun	PN	ብባንኪ/bbanki,
13	Phrasal Pronoun	PPn	ንባዕልና/nbaOlna፣ ንዓካትኩምን (nOekatkumn)
14	Phrasal Adjective	PAj	ንሰለማዊ /nselemaw፡ ንማሕበራውን/nmaHberawn
15	Phrasal Verb	PV	ንዝተሓቱ /nzteHatu፣ ንምቅላልን/nmqlaln
16	Phrasal Numeral	PNu	ንክልተ/nklte፣ንክልተን /nklten
17	Compound	Cp	Oamete-mHret(ዓመተ-ምሕረት) XelebegiO(
18	Contraction	Cn	‘mo(‘ሞ), anemo(አነ’ሞ)
19	Symbols	Sy	All symbols except the punctuation mark
20	Abbreviation	Ab	ዓ/ም (year)፣ር/መ (school head)
21	Foreign words	Fn	Symbols, English
22	Residual	Rs	Words not found in the corpus was assign this tag

Noun

As mentioned noun is a basic word class which is used to call people, place, things and abstracts. Therefore, all names used to call things specifically and collectively were included under noun. Moreover, noun can be classified in to proper noun and common noun; abstract noun and con-

crete noun. There is no clear way for making plural Tigrigna nouns. Tigrigna nouns are attached to different words to show emotion, action, relationship and so on. Therefore, it is difficult to call them as a noun instead it is preferred to use phrasal noun. For instances, let's elaborate on example: -

ሙሉጌታስ ክስዕስዕ ማዓት!MulugietksOsO maOat!

In the above example ሙሉጌታ is a noun and the suffix ('-ስ') added at the last of the noun is used to express performance of mulugieta in dance. Basically, the word is a noun but when we come to the work of part of speech tagging it is unfair to put them as a noun. Instead for this work purpose we put then as a phrasal noun. To make it concrete reason in the below example it share the same reason as in the above

mulugieta'ma nab srHu keydu:: ሙሉጌታ'ማ ናብ ስርሑ ከይዱ::

'ma is a short form of Ama. Separately, it doesn't have meaning, thus most of the time we found it as a suffix with other words as in the above case. It is not a conjunction, preposition but as in the above case it express the speakers answer for what is done by the subject, it also unfair to put them as noun. See also the below example, the word mulugieta is a noun where as the prefix **n-** shows that 'for' so instead of classified as a noun it is noun with preposition.

ንሙሉጌታ ቅርሺ ሃብዎ::nmulugieta qrSi habwo::

Generally all proper noun, common noun, abstract noun and concrete nouns are clustered in to noun word class excluding the aforementioned exception. Thus, all words classified under noun word class, should be tagged with 'N' tag label since we are using the tag-set at course-grained level. For example lets' see the below sentences

ሙሉጌታ ምስሑ በሊዑ። **mulugeta** eats his launch. (mulugieta msHu beliOu::)

Here in the sentence mulugieta (ሙሉጌታ) is the noun and will be labeled 'N' (ሙሉጌታ/N)

ዓዲግራት ዕብቅቲ **ከተማ** እያ::(adigrat is a beautiful city.) (Oadigrat Xibiqit ketema Aya::)

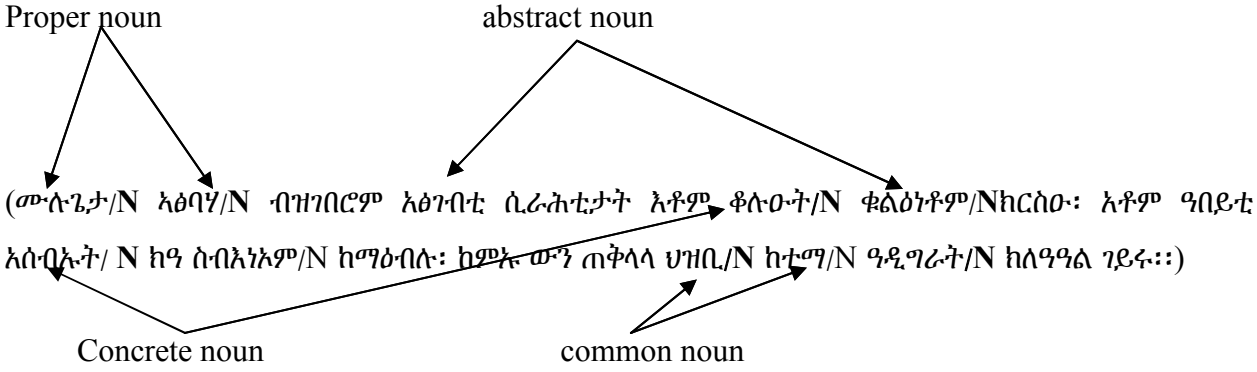
Here in the above sentence the word **ዓዲግራት** is a noun which was assigned to specific place and will assigned with the tag label 'N' so the assignment was held as **(ዓዲግራት/N)**

For instance lets' see the next example

mulugieta aXbaha bzgeberom aXgebti sraHttat Atom qolOut qulOnetom krsOu: Atom Oabeyti asebut kOe sbAnetom kemaOblu: kemu wn Teqlala hzbi ketema Oedigrat kleOaOal geyru::

ሙሉጌታ ኣፅባሃ ብዝገበሮም አፅንቲ ሲራሕቲታት እቶም ቆሉዑት ቁልዕነቶም ክርስዑ፡ አቶም ዓበይቲ አሰብኡት ክዓ ስብእነአም ከማዕብሉ፡ ከምኡ ውን ጠቅላላ ህዝቢ ከተማ ዓዲግራት ክለዓዓል ገይሩ፡፡

In the above sentence there are different kinds of nouns utilized for different purposes; even if the noun will possess different functionality for the purpose of this thesis all noun will assign a tag label called 'N'. So lets' tag the above sentence accordingly.



So accordingly all the nouns should tag with the tag label 'N'.

Pronoun: provides the same functionality like that of noun functionality provides. It is also used to create inference for in a give text document to refer the preceding sentence. As mentioned the most linguistic expert explore the uses of pronoun (Teffera, 1979), (Sahilu, 1998) and (Teklu, 2008) is as replacement of noun. i.e instead of using noun we can use pronoun. Thus, any word that replaces noun and utilized in the noun place is a pronoun. Therefore, all pronouns will tag with 'Pn' tag label. The same is true due to the morphological complexity of the language any pronoun that was prefix or suffice with other word and if the word can't be separate from pro-noun is excluded.

For instance lets' see the following Tigrigna sentences.

e.g.1

ንሱ/Pn አንበሳ ቀቲሉ። he killed a lion. (**nsu** anbesa qetil።)

From the above sentence lets' look at the word 'he' **(ንሱ)** refers to the man which was killed the lion; thus, the word 'he' was represent the man. Accordingly the word will be tagged as '**ንሱ**/Pn'.

e.g.2

ንሳቶም/Pn ወርቂ ምስ ሰረቁ ተአሲሮምሉ። (nesatom werqi ms serequ tasirom።) (As they stolen the gold they were jailed.)

In the above sentence the word class of word **ንሳቶም** (nsatom) is pronoun and it refers to group of people which was jailed as they suspected for the missed gold. This implies that a number of peoples was represented with the pronoun '**they**' and as we have mention in chapter 3 section 1 the word 'they' was represent for a group of women and group of man. Accordingly the word will be tagged with the tag label 'Pn'.

Adjective: as mentioned in the above section adjective will placed in front of nouns and will provides more powerful explanation to noun. So every word that modifies and explain the noun will be tagged with the tag label 'Aj'. However, it is impossible to say that all words come before Tigrigna noun are adjective, similarly it is also true everything that modifies nouns are not an adjective. Thus, all Tigrigna adjective is tagged as 'Aj'. But , If a word is adjective and the word is prefix of sufficing with other word and if the word is not separable from the adjective it not fair to assign the tag 'Aj' instead 'PAj' is preferable to this work purpose'.

እታ **ሓፃር**/Aj ጋል። (**HaXar** gal።) (The Short girl.)

ሽማግለ/Aj ሰብኣይ።(**Smagle** sebay።) the old man.

ፀሊም ወዲ።(Telim wedi)(black boy.)

Here the word (ሽማግለ:'(old)) is used to explain the status of the in which the man is old. Definitely the word will be an adjective and attached with the tag label 'Aj' so the output for the specified word is (ሽማግለ/Aj) this tag will be assigned for all words that explain and modify noun with the above exception. i.e, the word class doesn't include phrasal adjectives, because for this work purpose phrasal adjective will have their own class label (Tsegay, 2014).

Adverb: Tigrigna adverbs are words that qualify or modify a verb and other adverbs (Teklu, 2008). However, we can't say that all words that modify and qualify a verb are an adverb. The next two examples elaborate this issue in detail, the example are taken from the work of (Teffera, 1979).

A. Yesterday Mulugeta has gone to Adigrat. (ሙሉጌታ ትማሊ ናብ ዓዲግራት ከይዱ።።) (mulugieta tmali nab Oadigrat keydu::)
B. Kiflom dies by TB. (ክፍሎም ብዓባይ ስዓል ሞይቱ።።) (kflom bOabay sOal moytu::)

According to the above context in the above two sentences the word preceding (in example A (**ዓዲግራት**) and in example B (**ብዓባይ ስዓል**)) the verbs ((gone: 'ከይዱ' and die: 'ሞይቱ)) are not considered as an adverb because in the accordance of the sentence structure the two words has constitute their own word class. However, these words explains the verb in-terms of time and effect respectively in this manner we can consider them as adverbs. Thus, the words are nouns as a word class and can't classify as adverb.

e.g2

Sometimes Mulugeta was gone to his brother. mulugieta nab Anda Hawu HalHalifu ykeyd Ayu. (ሙሉጌታ ናብ እንዳ ሐዉ **ሐልሐሊፉ** ይከይድ እዩ ።።)

In this example the word (sometimes: '**ሐልሐሊፉ**') was an adverb and it explain the verb next to it. Basically it refers to the status of mulugeta when did he going to his brother (sometimes፡ **ሐልሐሊፉ**) accordingly the tag label that was attached with this word is 'Ad' so **(ሐልሐሊፉ/Ad)**.

So every adverb that constitutes the behavioral aspect of the Tigrigna language adverb was assigned the tag label 'Ad'.

Verb: a sentence without verb can't translate complete information (Teklu, 2008) and most of the time it was found at the end of the sentence. Accordingly all Tigrigna language word that is considered to be a verb will be attached with the tag label 'V'. See the next example

A) hiewan kdan HaXiba:: (ሄዋን ክዳና ሐፂባ።።) Hewan wash her clothes.
B) mkiele msHu beliOu:: (ሚኪኤለ ምስሑ በሊዑ።።) Mikiale eats his Lunch.

In the above example both words shaded in bold was resided in the verb word class. As we have mention in the above verbs are found at the end of a sentence as well as they reflects action, manner and so on in which the action doer possess. So as the subject in the first case (Hewan) doing something with her clothes which is **washing (ምሕፃብ)** of her close. Therefore, the word

Wash'**ሐፂባ**/V' was tagged with the tag label 'V'.

In this tag label all word which was include, show and has the same manner as verb in the word class verb was labeled with 'V'. For instance see the below verbs.

zTemete (ዝጠመተ): is a relative verb and for this work purpose it will attached with 'V'

Ayu (እዩ): is an auxiliary verb however for the purpose of this work it will be attached 'V'

nmrggaX (ንምርግጋፅ): is an infinitive verb however it was attached with 'V' for this work

Preposition: as we have explained in the above Tigrigna language preposition are words which cannot convey any meaning alone unless they are attached to other classes like noun, adjective, adverb and so on. For this work purpose we try to identify preposition as a general class. Let's explain the class with example:-

A) kebabi klte miti aTal teseriken:: (**ከባቢ** ክልተ ሚኢቲ ኣጣል ተሰሪቀን።።) around two hundred goats are stolen.
B) ab wSTi wenber alo:: (ኣብ **ውሽጢ** ወንበር ኣሎ።።) it is under the chair.

Preposition

As we have indicated with arrow in the above the two words (Around፡ **ከባቢ**፣ under፡ **ውሽጢ**) are prepositions. Accordingly the two words are attached with tag label 'Pp'. In this class all prepo-

sition was found and they will attached with the respective tag label as (ከባቢ/Pp), and (ውሽጢ/Pp) respectively as to the above two sentences.

Punctuation mark: all punctuation marks are tagged by 'Pu'.

e.g1

A) mulugieta nabey keydu? (ሙሉጌታ ናበይ ከይዱ ?) Where did mulugeta Go?

Here the punctuation mark was attached with the tag label 'Pu' so (**?**/Pu)

Numerals: like Numerals for Amharic, Afaan Oromo and other languages Tigrigna numerals can be cardinal, ordinal, decimal and special numerals and which was tagged as 'Nu '.

Conjunctions: are words used to connect words, phrase, clauses or sentence. For this work purpose Conjunction that can be used as a separate word is tagged with the tag label 'Cj '.

Interjections: are words used to express feeling and emotion. Let's ' see the next example.

A) **bravo**! zwedy blaOeyo:: (ብራቮ! ዝወደይ ብልዓዮ::) Bravo! My son eats it**.**

In the above sentence the speakers uses the word bravo (**ብራቮ**) to reflect his/her strong feeling to his/her son. Accordingly the word bravo (**ብራቮ**) was tagged with the label 'Ij' so (**ብራቮ**/Ij)

Phrasal noun: this class label includes all Tigrigna

A) noun attached with conjunction and can't be separated
B) noun attached with preposition and can't be separated
C) noun attached with preposition and conjunction and can't be separated was attached with the tag label 'PN' lets' see with more detail example

A) kalayu bbanki zsededo genzeb teseriqu: (ካላዩ ብባንኪ ዝሰደዶ ገንዘብ ተሰሪቁ::) the many sent by kalayu through Bank was stolen.

In this example the word bank was a noun and it was presided by the preposition (through: **ብ**) so it takes the form Noun with Preposition and it directly accepts the rule for the Phrasal noun and will assign the tag label 'PN' so (**ብባንኪ**/PN) will be the final output.

B) mulugetan kalayun teXarerti Ayom:: (ሙሉጌታን ካላዩን ተፃረርቲ እዮም::)

In this context the word mulugieta is a noun and it was attached with conjunction to refer the relationship among the two nouns (mulugieta **and**kalayu). Accordingly the word (mulugietan and kalayun) was tagged with the tag label 'PN'

Phrasal pronoun: all pronouns that was attached

A) pronoun with preposition and can't be separated from
B) pronoun attached with conjunction and can't be separated
C) pronoun attached with conjunction and preposition and can't be separated was labeled with tag label ''PPn'

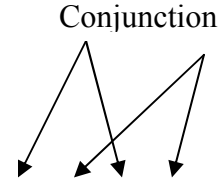
Phrasal Adjective: all Adjectives that was attached

A) Adjective with preposition and can't be separated from
B) Adjective attached with conjunction and can't be separated
C) Adjective attached with conjunction and preposition and can't be separated was attached with the tag label ''PAj'

Phrasal Numeral: all Numeral that was attached

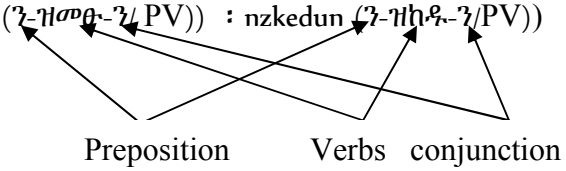
A) Numeral with preposition and can't be separated from
B) Numeral attached with conjunction and can't be separated
C) Numeral attached with conjunction and preposition and can't be separated was attached with the tag label ''PNu' verb

Phrasal Verb:all Verbs that was attached



A) Verb with preposition and can't be separated from (b-mkad: (ብ-ምካድ))
B) Verb attached with conjunction and can't be separated (keydun rekibun (ከይዱ-ን ረኪቡ-ን))

C) Verb attached with conjunction and preposition and can't be separated was attached with the tag label ''PV' (nzmeTun:(ን-ዝ*መ*ፁ-ን/PV)) : nzkedun (ን-ዝ*ከ*ዱ-ን/PV))



Compound words: as the other language Tigrigna language has also compound words that can be reflecting different meaning than the individual words (Teklu, 2008). Compound words are words that integrate two or more than two words with a single component and inhabit different meaning than of the two integrated words. For this work all words create different meaning than the individual word through compounding was attached to the tag label 'Cp'.

Abbreviation: in Tigrigna language Abbreviation was written with forward slash, colon, space in between the characters like (ዓ/ም፣ ዓ-ም ፣ ዓ*ም*፣ (year) ፣ ር/*መ* (school head)) accordingly all the abbreviation was tagged with the tag label 'Ab' like (ዓ/ም/Ab)፣ (ር/*መ*/Ab)

Symbols: all symbols used in the text corpora was tagged with the tag label 'Sy' like (©/Sy)

Foreign words: all non Tigrigna language words was tagged with tag label 'Fn' like (English/Fn), (mind/Fn)

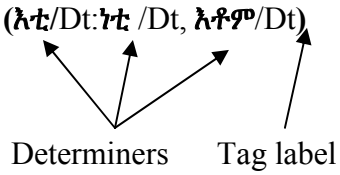
Determiner: determiners are words that occur together with a noun or noun phrase and serves to express the reference of that noun or noun phrase in the context. Accordingly for this work purpose all Tigrigna language determiners were tagged with the tag label 'Dt'.

እቲ መንበር ኣልዕሎ። (Ati wenber alOlo።)

ነቲ ማይ ከደኖ። (neti may kdeno።)

እቶም ኣሰብኡት ናብ ቀበሌ ከይዶም። (Atom asebut nab qebelie keydom።)

All the words shaded in bold were determiners and attached with the tag label '**Dt**' accordingly



Residual words: refers to all words which are out of vocabulary (i.e. unseen and most words that create ambiguity one tag from the other). Accordingly the tag label for those words is 'Rs' like if any word not found in the vocabulary then the label will be (unseen word/**Rs**)

CHAPTER FOUR

MODELING AND DESIGN TIGRIGNA POS TAGGER

4.1 Overview

The task of tagging can be explained as predicting values of a missing data onto a give data table column. Thus, we can assume the missing value can be a tag and the data column is a candidate token. To overcome the problem different kinds of approaches and techniques was proposed and implemented for different languages.

As explained in chapters 3 section 1, part of speech tagging is simple but requires careful attention while designing. The designed and trained tagger can use as first stage for other high level language processing tasks. Even though, several taggers are designed and proposed to other languages they cannot directly implement for Tigrigna; due to the difference in nature and morphological complexity of the languages. Absence of Tigrigna PoS tagger hinders researchers not to do other high level NLP applications and innovation for Tigrigna. All problems start from the very beginning of readymade language resources and references used as a starting point to perform different language technology tasks (Tedla, Kazuhide, & Ashuboda, 2016). However, there is an attempt on doing part of speech tagger for Tigrigna 6 years ago by Gebregzabiher (2010). Beside of designing the tagger the researcher fails to build robust and efficient tagger. The reported tagger was face lack of performance since it was only trained and tested with a single genre (news corpus). The problem can be seen as a boatel neck of the tagger. Even if it scores high accuracy as reported but it fails even closer result while it feeds with dataset from other genre. The approach is data driving approaches requires high amounts of dataset to training the model. Another drawback is the window size used to build a hybrid model. It only uses bigram of word if initial tagger fails to fulfill the specified fixed threshold value. So in considering this and other constraints in the first PoS tagger, designing and evaluating a robust and efficient tagger is required. Throughout, this Section, clearly explain the design issue and techniques used for Tigrigna tagger.

4.2 Rule Based Tagger

As discussed in chapter 3 section 3.3.2 Rule based tagger uses a set of rules as a main component of the tagger. Rules can be drawn either manually, or using example data automatically. Deducing rules manually are time, labor intensive and it requires linguistic expert that has deep knowledge of the concerned language. As a result, it provides clear information about the available dataset and permissible rules help the automatic tagger to deduce correct labeled datasets (Bril, 1995). However, deducing rules automatically is simple and easy but it may result rules that don't represent morphological and syntactic property of the language. For this work purpose deducing rule automatically through machine learning algorithm was utilized. As clearly depicted and documented, Brill tagger is a transformation based tagger that performs very well, but uses only a tiny fraction of the space required by the nth-order stochastic taggers. The general ideas of the tagger are very simple; it uses a set of rules to tag unlabeled dataset at the same time it may learn some new rules. The rules are implemented in the temporary corpus data to improve the tagging accuracy. The process continues until no rule found to improve temporary corpus (Addisu, 2016).

TEL works in two phases; first the tagger tags input tokens with selected initial state annotator; then a set of transformation rules are applied to tagged temporary corpus to make temporary corpus closer to reference corpus (Eric, 1995).

The initial state annotator is used to assign initial tag to un-tagged data inputs. Afterward, a set of transformation rules were applied to learn lexicons and context. Then, the produced labeled data is compared to reference corpus (gold standard corpus) as depicted in Figure 4.1.

During learning phase, a transformation rule score high accuracy was returned and added to ordered transformation rule list. Then, the temporary corpus is updated by applying the learned transformation rules. As a result, the designed model uses the stored transformation rule deduced from the training phase to tag unseen text. In the next section we will explain how this can be applied to Tigrigna text documents.

The tagger was learned by detecting errors. TEL begins with un-annotated Tigrigna text document as input. Since the documents are normalized in prior, each document was passes through

preprocessing phase to make it suitable for the model, In case if there is any un-normalized data-set is found during this phase the document was split into sentence level using Tigrigna sentence delimiters and each member of the sentence is treated separately except null value members. The normalization component of the tagger uses different functions to normalize the input dataset. First the entire dataset were processed at sentence level using Tigrigna sentence delimiters, considering source of our corpus different sentence delimiter was found in the corpus (specifically the dote (.) or called (arbaOte neTbi)) to create common punctuation for the tagger we compile all different dots to a single dot (።) and all the other sentence delimiters are used as they appear in the dataset. Afterward, we split the sentence in to token levels to see whether the token needs further normalization or not sees corpus preprocessing in chapter 5 sections 5.2.2. In addition to this to manage all foreign words with the file we create a regular expression pattern to eliminate foreign word as much as possible from the files even if there is a tag level for foreign words. After both sentence to tokenize and word tokenize methods were applied the selected initial state annotator assigns initial temporary tags in same fashion. Output of initial state annotator is a temporary corpus (TCo). For each time the temporary corpus is passed through the learner, the learner produces at least one new rule. Then the rule is applied to the entire temporary corpus for error detection and correction. So if the rule scores high accuracy than preceding rule and specified in the threshold value limiter the rule will be stored and the initial state temporary corpus (TCo) is amended (TC1) accordingly. However, if a rule is scoring worse, then, the rule will discard another rule is detected and applied to the temporary corpus. Only a single rule that improves the annotation of temporary corpus compared to reference corpus (manually tagged) is added to a rule list. Using the process the learner produces an ordered list of transformation rules. Finally, these rules can be used to tag new un-annotated corpus. The tagger uses TEL once in a lexical module to derive lexical rules, once in a contextual module for deriving contextual rules.

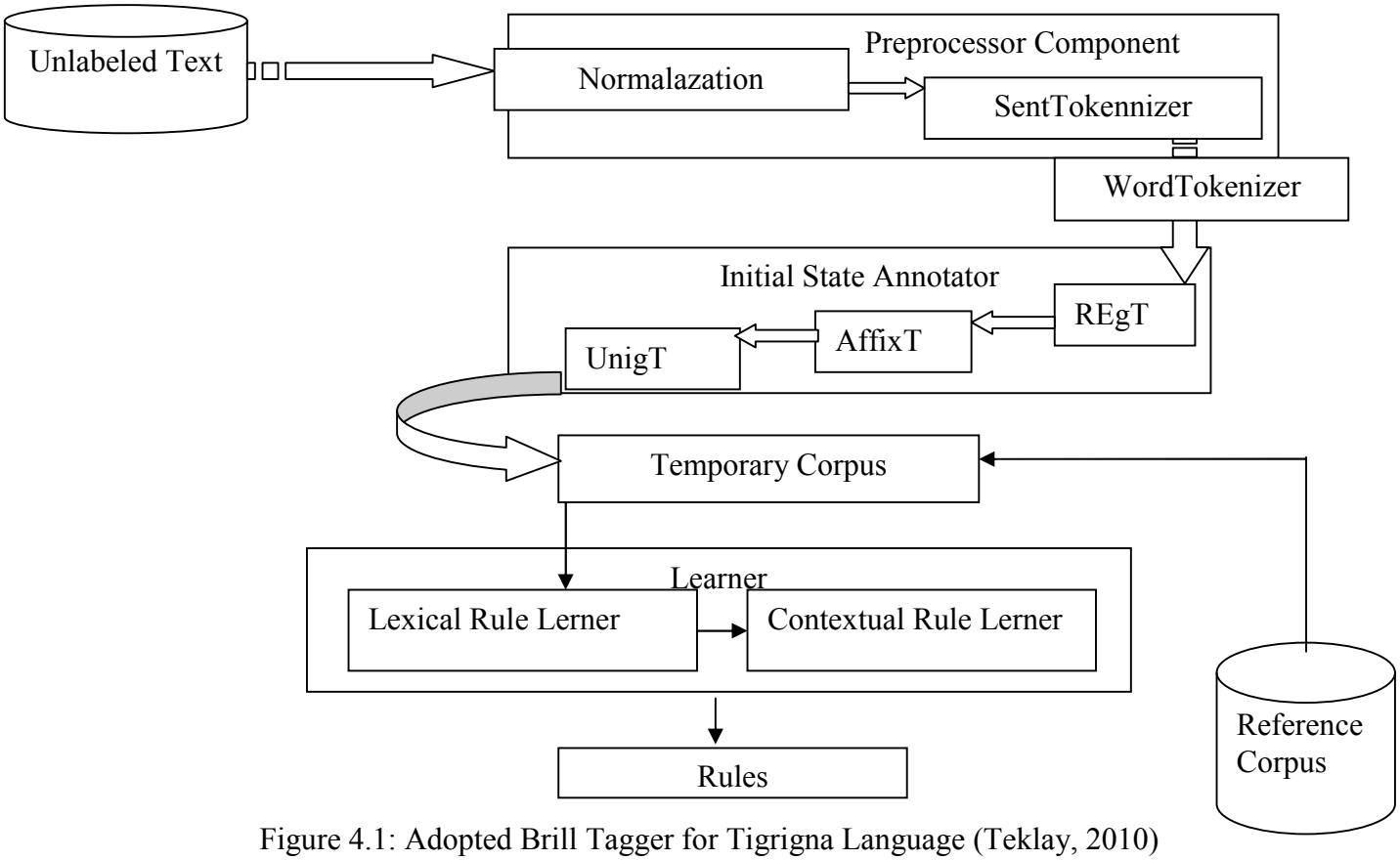


Figure 4.1: Adopted Brill Tagger for Tigrigna Language (Teklay, 2010)

Both modules (Lexical and Contextual) uses two corpora a goal corpus (manually annotated corpus), and a temporary corpus (corpus created during initial state annotator) as depicted in Figure 4.1. In lexical module reference corpus contains a list of words with information about the frequencies of tags and the temporary corpus consisting of the same list of words as in the goal corpus, tagged in some fashion. In contextual learning module a goal corpus is a manually annotated text while a temporary corpus consists of the same text as in goal corpus but with different tags. Brill tagger rule consists of two parts; the condition (trigger) and resulting tag. The rules are instantiated from a set of predefined transformation templates. They contain instantiated variables and are of the form:

If Trigger, then change the tag A to the tag B. In which an and B are two different variables where if the rule triggers and if the assigned current tags 'A' satisfies a condition on the active

token, then the rule replaces current tags with resulting tag B. Therefore, a set of all permissible rules are generated from all possible instantiates of all predefined templates.

4.2.1 Initial State Annotator

After all Tigrigna document was passes through preprocessing component unseen dataset was present to initial state annotator. As explained earlier initial state annotator assign initial temporary tag to candidate token. Initial state annotator stipulated to Brill Tagger can range from simple annotator to complex one. For example Gebregzabiher (2010) uses default, bigram and trigram, Mekuria (2013) uses default and unigram tagger as initial state annotator. In both case they use data driving tagger as a backoff method which is possibly lead to the concept of stochastic. i.e. by their nature stochastic taggers require large amount of data to disambiguate and better result . Another thing as they require large amount of tabling in training phase they are too large in size. In contrast to Brill tagger uses a large number of templates, using bigram and trigram as initial state annotator with Brill tagger increase training time and may also increase possibility of tagging ambiguity. Most researchers use default tagger as initial state annotator to assign same tag to every candidate during this stage. The tag label uses as a default tag can be the most frequency tag in the entire corpus (Gebregzabiher, 2010; Mekuria, 2013). However, using default tagger as initial state annotator may result inconsistent tag assignment to unknown words. The selected tag for default tagger can also be based on nature and complexity of the intended language in consideration of frequency of tags (Gebregzabiher, 2010). Having the above concept, to the best of the researcher different source of knowledge was used to combine initial state annotator with Brill tagger for this work instead of using stochastic annotator alone.

As explained to combine initial state tagger annotator we depend on the sources of knowledge that can help us decide the best sequence of suggested tag for a give token. Thus, different source of knowledge integrated into our initial state annotators are:-

- Patters that exclusively identify word class category (Patterns using regular expression tagger)
- The morphological form of the word (suffixes, prefixes) (using Affix Tagger)
- The statistical appearance of the word throughout the corpus (Using Unigram Tagger)

On the bases of the above sources of knowledge, we then combine a sequence of taggers that uses the aforementioned sources of knowledge. Detail description on how we combine this knowledge was presented next.

4.2.2 Models Combination

To understand the difference, consider that a tagger may be given the option to “pass “on difficult questions. i.e when a model is designed carefully it only commit to an answer if it is sure of the answer only Like the pattern tagger (regular expression tagger). Such a tagger can produce high accuracy, but may run the risk of low accuracy for unseen candidate with respect to reference dataset. In contrast, a risky tagger will suggest an answer even it is not sure of the answer. Such a tagger could reach high accuracy for unseen data-set. However, the risk of accuracy in relation to the closeness to reference dataset is low. For in-stance unigram tagger, which only assign tag based on the word/tag frequency? Thus, combining different tagger into one model as a backoff tagger is expected to perform better than single approaches (Bird, Klein, & Loper, 2005). Beside that it also expects to handle unseen data in training. Naturally, it makes sense that combining two sources of knowledge produce better decision.

According to the above concept the way we follow to balance with different sources of knowledge is feeding output of one tagger to the next tagger considering better accuracy. i.e. model that looks at morphological structure of words and one that look at context of words use completely different knowledge sources .

For this work purpose to combine taggers we use a chain of tagger and order them in a sequence of backoff module. The backoff chain method is the most “picky " start. Using default tagger is an obvious solution to improve tagger accuracy; however, we prefer to start with Regular expression tagger. The main reasons were using simple pattern combination we can handle the most important property of Tigrigna. Then output of RegTagger () was passed to AffixTagger () and output of AffixTagger Was pass to Unigram Tagger (). Finally, Brill Tagger () produces final labeled dataset. The ideal logic behind combining tagger were simple if initial tagger decides to " pass " , and then it asks another , less picky tagger to contribute over result produced . The chain can be extended as long as the researcher needed and complexity of the tagger. To perform these

tasks we use NLTK module. Tag Sequential taggers. The interface sequential backoff Tagger inherits from the Tagger interface. For this matter the first three tagger is classified under sequential tagger (i.e tags words with left to right, in order, and at each step, they can look at the decisions made in the past words with the sentence). The first state tagger assigns tag to words, and if the initial tagger is not sure about the tag of the cur-rent active word, it passes to next tagger for further investigation.

As mentioned, Regular expression tagger looks at morphological structure of words. To do this, first, we set different patterns using the morphological property of Tigrigna in NLTK. Regexp Tagger to handle the candidate tokens satisfies the expressions. Once candidate token satisfies the expression regular expression tagger decides the tag. Else it tries the next tag. In relation to the implementation of the other tagger the implementation of the Regexp Tagger is quite simple. However, due to morphological complexity of Tigrigna we face to different questions on sequence of the pattern: mainly the two most common questions we should to answer are:-

- How can we select better regular expressions?
- Order of regular expressions patterns?

However, to select the best regular expression those feet with the morphological complexity of the language we perform a number of discussions about Tigrigna language experts. So we decide to select the order for on our discussion. Basically, we do our best to come up with better result and accuracy. But one thing should to be considered here is morphological complexity of the language will hinder us to do more things. For example as English there is no clear way of identifying class with words using single patterns like (identification of Adjectives or adverbs.). So as a best solution to answer the above questions we use the Affix Tagger to learn these parameters from a training corpus. Affix Tagger is a trainable tagger that attempts to learn morphological property of word. It only looks at the last letters in the words in training corpus, and counts how often a word suffix can predict the word tag as put in the default Affix Tagger. For the purpose of this paper we configure and customize Affix Tagger to look at Suffix and prefix of words to fit Tigrigna language word morphological property. In other words, we learn rules of the form ('.*xyz', PoS) or in the reverse direction (' xyz .* ', PoS). Affixes are identified based on the passion for the stems they attached. If they attached preceding the stem they are called prefix and if

they are attached following the stem they are called suffixes. Tigrigna is very rich in affixes (Tewelde, 2002). So Regular Expression tagger helps Affix Tagger if we use it as a backoff instead of the Affix Tagger alone. On the other hand the Affix Tagger Also utilized as the backoff tagger for the unigram tagger. The unigram tagger was used as a backoff tagger to Brill initial state annotator. So it performs high accuracy as in the state of the art in tagging for resources limited and morphologically reach languages. The detail architecture of the combined model was present as follows:-

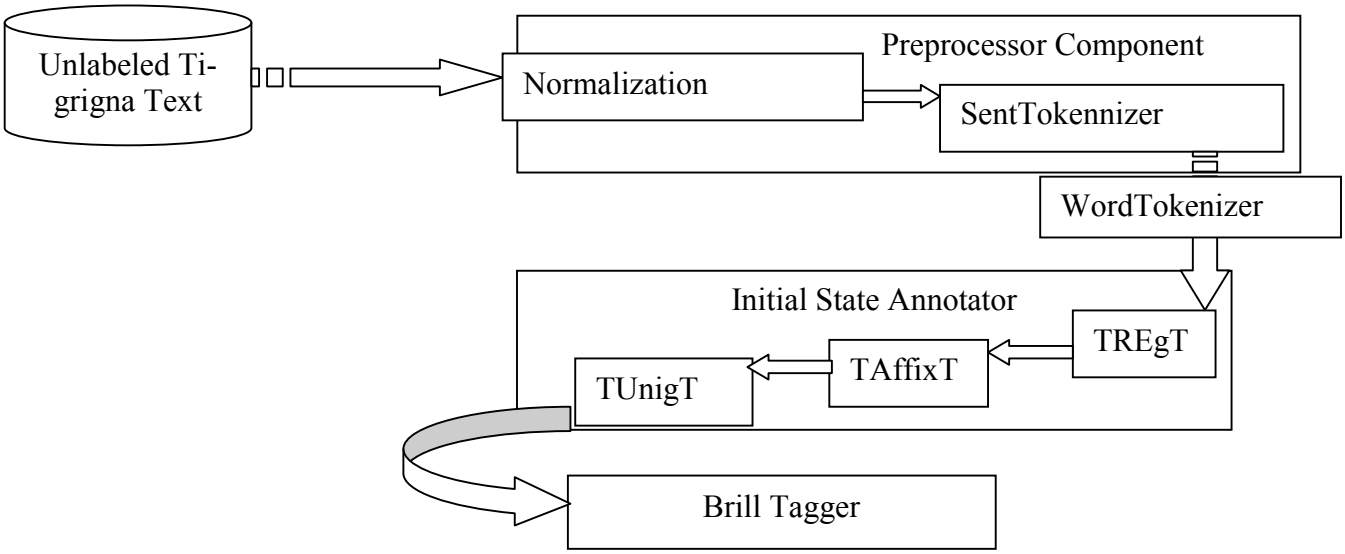


Figure 4.2: Combined initial state Annotator Architecture (Bird, Klein, & Loper, 2005)

4.2.3 Learning Lexical Rules

The ideal goal of the lexical module are used to produce lexical rules (Brill, 1995) that can produce the most frequent tag for candidate token considering all texts in the corpus . However, the problem is determining the most likely tags for unknown words, given the most likely tag for each word with a small set of corpus. In this Context the learner is used to drive lexicons and rule that assigns the most likely tag for a given word that may or may not be seen in the training corpus. Statistical method is used to compute the lexicon and it contains list of words information along with its most frequent tag. This information is used to tag untagged tokens at least seen one times in training corpus.

The learner takes temporary annotated Tigrigna text passes through initial state annotators. Based On lexical rule learner template, best permissible transformation lexical rule will be returned. Best rule meant a rule that gives better resemblance between the reference texts when applied to temporary corpus created during run time (Gebregzabiher , 2010), (Mekuria , 2013). Score computation for the Best rule can be computed as: For each tagged tokens in a temporary corpus a rule gets a score for that token by comparing a change from current tag to resulting tag with respect to the token in the reference dataset. The Score Returned for a rule may be.

- Positive (The rule changes tag of a word from incorrect to correct), or
- Negative score (change tag of a word from correct to incorrect) or
- It can be zero (Condition not satisfied).

The Rules that have best score are applied to temporary corpus and the temporary corpus is amended accordingly. Afterward, a rule that returned bests scored and amend the temporary corpus was added to sets of rules. The process continues in the same fashion until no rule found that further improves a tag of the temporary corpus. Rules are ordered, i.e. the last rule is dependent on outcome of the earlier rules (Mekuria, 2013).

The lexical rule learner deals with morphological property to drive a set of all permissible lexical rules. Some of the transformation templates that are used in lexical rule learner are given below.

1. Change the most likely tag to Y if the current word has suffix X.
2. Change the most likely tag to Y if deleting/adding the suffix X, |X|<4, results in a word, |X| is length of x.
3. Change the most likely tag from X to Y if deleting the prefix (character) X, |X|<4, results in a word, |X| is length of x.
4. Change the most likely tag from X to Y if word W ever appears immediately to the Left/right of the word.
5. Change the most likely tag to Y if the character Z appears anywhere in the word.

In the adapted Brill tagger for this work, considering nature and morphological complexity of Tigrigna texts the original Brill tagger Template numbers 2 and 3 are used as in the work of

(Gebregzabiher, 2010) and (Mekuria, 2013). The template can be interpreted as adding or deleting up to 4 characters from prefix or suffix in a candidate token. Most Tigrigna words add up 1- 4 prefix and 1- 4 suffix. In addition to this all phrasal word classes are considered in this context. For instance in example **A** the root word is **ደዊልካ** (dewilka); however it is preceded by 3 prefix and 2 suffix as in the example shown, in this case the pattern match mechanism is utilized because mostly they are rare in the collected dataset. However, example B, C, and D**,** are frequent word with corpus and we set the values as in the above rules.

1. እንተደዊልካለይ(Antedewilkaley)"if you call me"እንተ (Ante)-ደዊልካ (dewilka)- ለይ (ley)or እንተ(Ante)-ደዊልካለይ(dewilkaley)
2. ኣይከደን(aykeden)(he is not going.)ኣይ-ከደ-ን
3. ሰለዝመፀ(slezmeXe)(as he come.)ሰለ-ዝ-መፀ
4. ከምዝኮነ(kemzkone)(as done.)ከም-ዝ-ኮነ

4.2.4 Learning Contextual Rules

The contextual module is used to produces contextual rules. It is used to disambiguate and attempted to better performance. After the tagger learns the most likely tag to each token in training dataset, the contextual rules are learned. Then, the contextual rules are induced on the basis of initial state temporary corpus (TCi) and the lexical information collected during lexical rule Lerner. During first state, a learner generates a set of permissible rules for all possible predefined contextual templates. Then, it computes score of each rule for word level and it takes rule for the highest score. The rule selected with the highest score put on rules component as an output. Afterward, the learner uses the rule to generate TC1 (temporary corpus generated after rule one was applied to initial temporary corpus TC0), so learning continues until no rule found that modify the temporary corpus compare to reference corpus. A score of every permissible rule is computed as a rule is applied to the TCi. For instance

if rule 'R' is in component of permissible rule and if R is applied to a word (W) which is already in TC_i then score for rule 'R' calculated as:

For each word 'W ' in TCi, the contextual rule learner computes score of R on the word W. Then, scores of all words with TCi where the rule is applicable are added and result is total score of rule R. This can be done by comparing the tag of the words with TCi with the correct tags of the words with the goal corpus (Reference corpus) . If R is applied to word 'W' thereby correcting an error, the score for W is +1.

If applying the rule' R' to the word Wi and an error is produced then score of Wi will be -1. In all other cases, score of particular word Wi is 0. Therefore, total score of rule R was computed as.

$$\text{Score(R)} = \sum \text{number\_of\_errors\_corrected} - \text{number\_of\_errors\_corrected}$$

As mentioned earlier the main goal of contextual rule learners are to generate a set of permissible contextual rules. The sets of rules generated during contextual rule learner are totally different from sets of rules generated during lexical rule learner since they use different predefined transformation templates. The trigger for this case depends on the context of a word but no on morphology of a word. Some of the triggering templates are:

1. The preceding/following word is tagged with X.
2. The preceding/following word.
3. One of the two preceding/following words is tagged with X.
4. One of the two or three preceding/following words
5. One of the two or three preceding/following words is tagged with X.
6. The preceding word is tagged with X and the following word is tagged with Y.

4.2.5 TEL Rule

As explained earlier rules are considered as the main components of rule based tagger. The rules has sub-component with two main parts namely triggers (condition or current tag) and rewrite (resulting tag). Figure 4.3shows the main parts of a rule.

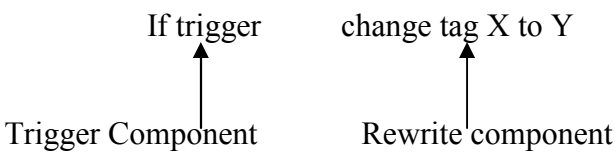


Figure 4.3: Main Part of Brill Rule (Mekuria, 2013)

Once the adopted TEL approach trained and learns a set of lexical and contextual rules using lexical and contextual rule learner sub-components of the rule based tagger store each learned rule within a file. In addition to these, the rule-based component builds a lexicon to tag words that are seen at least once during training time. All rule and dictionaries are stored in a file. As a result, rule based component becomes a trained model able to tag un-tagged Tigrigna texts using the stored information during training. Graphical representation of rule-based component of Tigrigna PoS tagger is shown in Figure 4.1.

Generally the lexical and contextual rule of Brill tagger learns rules in training. To tag a new text, Brill tagger uses rules learned from training corpus during the learning phase. The rules are applied to new un-tagged text and we get labeled text as output. First using the lexical rules is applied and then contextual rules are applied to look into the contexts of a word with the tagged temporary corpus and try to improve performance of the tagger. Finally labeled Tigrigna text is output from the tagger as in the example.

ኢኮኖሚ/N ዛበሌ/V ዮርፉ/N ህወሓት/Ab ህዛቢ/N ትግራይ/N ዮጋጊሙ/N ዖዛይሞ/PV ምቁር/Aj ዮርፉ/N እዩ/V ።/Pu ትግራይ/N ዒዴና/N ወርቂ/N ንኽትሸሆም/PV ካብቲ/Dt ብኣዔፅምትን/PN ዮምን/N ዮቃ/N ዛተነዴቐ/V ስርዒት/N ትፅቢት/N እትገብሮ/PV ብ▯ሕ/Aj ነገር/N እንተሃሆወ/V ካብዘአም/Dt እቲ/Dt ቀንዴን/PAj ወሳንን/PAj ጉዔዜ/N ዔብየት/N ዛኾነ/V ምስፈሕፈሕ/V ኢንቨስትመንት/N እዩ/V ።/Pu

4.3 Averaged Perceptron Tagger

In machine learning, perceptron is an algorithm based on supervised learning used as binary classifiers. It initializes all parameter vectors to zeros, and updates the vector by decoding the training examples (Zhang & Clark, 2010); i.e. it is a function that can decide whether an input belongs to one class or another. For each example, output of the model is compared with reference dataset. If the output is correct, no update is performed. If the output is incorrect, then pa-

rameter vector is updated by +1 to a true tag in the reference example and penalize the parameter vector leads to incorrect output. Perceptron algorithm can be defined as linear classifier (classification algorithm that makes its predictions based on averaged weight associated with features). The algorithm processes elements of the training set one at a time sequentially. As stated, averaged perceptron is a machine learning algorithm based on averaged weight prediction of part of speech tag as implemented in NLTK (Matthew, 2013). Simply perceptron is an algorithm used to map an input x to an output value f (x).

$$f((x)) = \begin{cases} 1 & if\ w.x + b > 0 \\ & otherwise\ \ 0 \end{cases}$$

where w is the current weighted value, and is the dot product of the current weight and the input x. afterward, the sum of the input weighted values and the current weight is computed as a dot product of where x_i stands for the number of inputs to the perceptron and w_i stands for the current weigh. 'b' is the bias added to the total sum of the weighted input. The bias shifts the decision boundary away from the origin and does not depend on any input value.

A detail implementation of averaged perceptron tagger for Tigrigna is present as follow:

Assume that we have a table containing data in which the last column value of the table is missing. Then if our interest is to find out the missing value of the data onto the last column then we can create correlation between the columns of the tables and then we can figure out the missing value of the data column. In our case the missing column can be seen as a part of speech tag for word and the data can be the current active token and the predictor column can be at the position of $word_{i-1}$ or $word_{i+1}$. Thus, for this kind of problems by applying averaged perceptron tagger we can solve it easily.

Diagrammatically general architecture of the adopted averaged perceptron model is depicted in Figure 4.4.

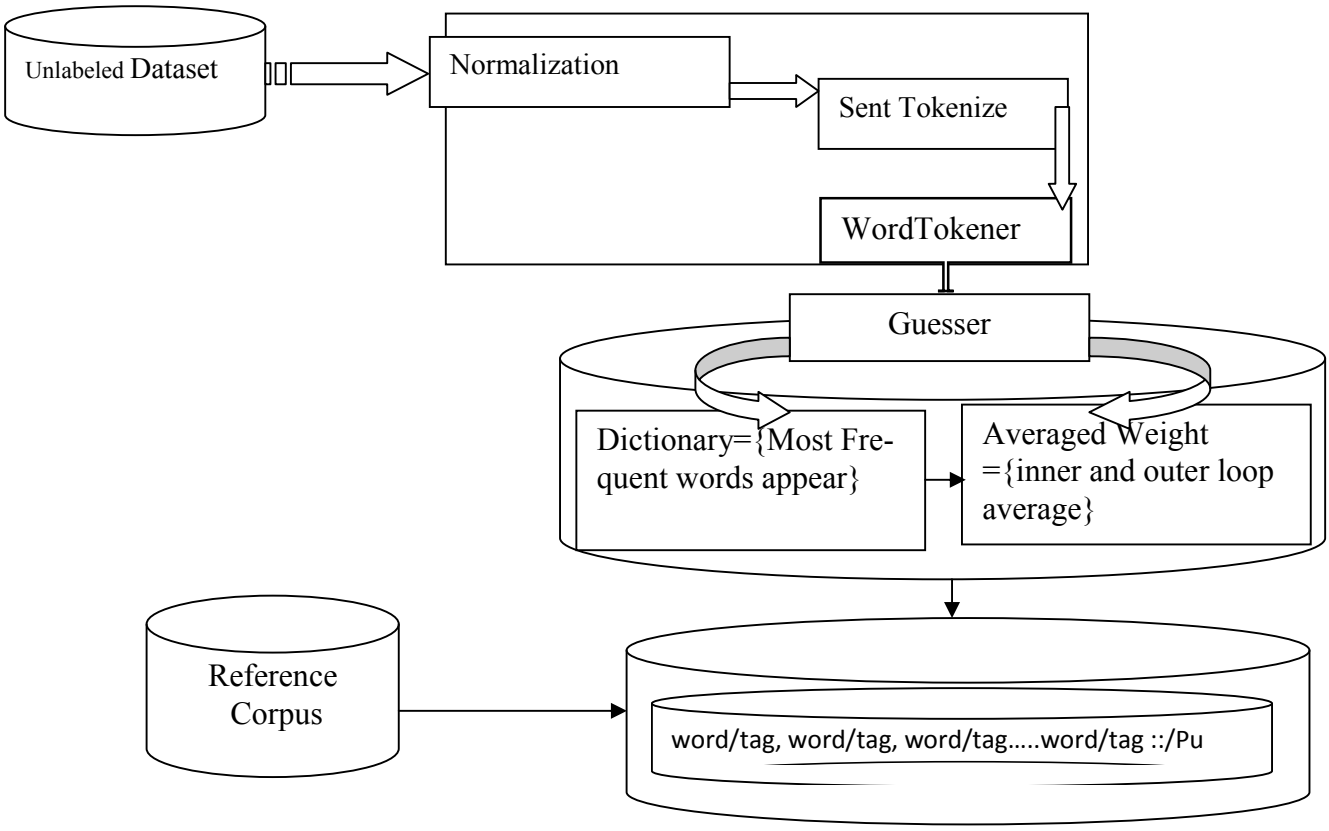


Figure 4.4: Adopted Averaged Perceptron Tagger (Matthew, 2013)

4.3.1 Prediction of PoS Tag

There are two main concepts when we come to averaged perceptron tagger tag prediction function. Tag prediction using the most frequent word with frequent word dictionary, and tag prediction using feature functions training Phase. During prediction an empty dictionary that maintains weighted value of feature/class pair is created. The empty dictionary helps us to map value of each data. The data-structure weight is a dictionary of dictionaries, to associate weight for each feature/class pairs. Usually members of the dictionary are sets of data input and feature pair with none zero value. Only tokens not in frequent words dictionary are considering during the phase. All features of each and every candidate token are considered as the main component to compute the current weight. Afterward, sum of the dot product of current weight and input weight was computed. Sum weight can be computed as:-

$$y_{i=f[wi.xi]=f\left[w_{ot}+w_{1t}(.)_{x_{j,1}}+w_{1t}.x_{j,2}+w_{1t}.x_{j,2}+w_{1t}.x_{j,4}+w_{1t}.x_{1,5}.....+w_{1t}.x_{j,n}\right]}$$

Where y_i refers to the sum weight of the input vectors, and we are predicting in the j^{th} time; For instance, if this is our fifth prediction j is 5, so our prediction looks like

$$y_5=f[w_0(t)+w_1(t).x_{5,1}+...+w_n(t).x_{5,n}]$$

$w(t).x_j$ is the dot product. It means we multiply each input weight with the current weights. So each input multiplies by a feature weight. Then, all computed weight passes for computation of averaged weight. As stated the main advantage of this model is it only stores averaged weighted values and ignores all intermediate values. Therefore after all values are averaged in some fashion the averaged weighted values are returned as a final weight for training dataset and unseen dataset. Based on the training dataset and the features of candidate token tag contains the maximum averaged weight are returned. Basically, there is no minimum or max threshold value to handle the weight of the token just compute the average and then return the max averaged weight as score value indicates the best tag. Simply this can be considered as one drawback of the tagger. That means, even a tag is incorrect tag label for a candidate token if it scores high weight it is assigned as a final label. This is one concept that's way we are inspired to integrate with the rule based tagger.

After all weight is computed and summed a candidate sequence of tag label are suggested for a candidate token. However, from the suggested tag label only a tag with a maximum weight is return.

4.3.2 Learning Weight

During learning phase our interest is to find a maximum averaged weight of a tag from a sequence suggested tags. By taking the maximum averaged weight helps to predict the best class label of the candidate token as comparing the label of the tag in the reference dataset. So on the bases of our initial concept we start with an empty weight and we iteratively store the averaged weight to an empty dictionary that maintain the weighted average of the feature/class pair. The weight computation was performed in three main parts.

- Examine the feature/class pair
- Assign value of PoS tag according to weight gained during feature weigh analysis
- If the assignment is wrong then the tag leads the algorithm to false prediction was penalized by one and the true tag is increased by one value in advance.

The steps are performed again and again until the correct tag is found. If variable 'i' is in the iteration range and if the feature and the true tag are in training dataset then the predicted value is returned. Otherwise, first we guess a tag according to averaged weight and if the tag is different from the true tag in a reference dataset, the assigned averaged weight is penalized by one and we add one to true tag weight as in the above, and then step number two is continue.

4.3.3 Averaging the Weight

In this phase all computed weight are not returned as final weighted values of a candidate token to predict a tag label. Instead averaged weighted values were computed. The computed weight should return as an averaged weight and all intermediate weight is ignored. This is a simple and perfect advantage of the algorithm. As a simple computation of averaging for a given input:-

avg $= \frac{sum\ of\ inputs}{number\ of\ inputs}$ or simply having the input values range from x_1 to x_n then average value of all inputs can be computed as

$$avg = \frac{(x_1 + x_2 + x_3 + x_4 + \cdots + x_n)}{n}$$

During learning phase we have two loops an inner loop and an outer loop. So the average value was computed with all the inner and outer iteration. In our case the iteration number is valuing ranges of 1 to 10. Based on the inner and outer loops we'll average across all values for each weight.

Finally we divide the computed average by the total number of iteration. We want the average of all the values of the inner loop. Assume that we have 3 examples, and we train for 2 iterations, thus, we'll average across 6 values of each weight. Basically our interest is not in the final

weighted value returned; instead we should give more attention on how to compute the average and concern on the unchanged values. So we will maintain all unchanged vales to one place with their time stump and we truck on the accumulator that maintains the average weight value of each and every feature/class pair that has a chance of amendment. Now we can make a fast and forward change update to the weighted average simple and easily. And we can maintain the un-changed values too.

4.3.4 Training the Algorithm

Training the algorithm takes two advantages. First all sets of word/tag occur more than 20% is accessed and stored in a separate dictionary. The ideal logic are if a candidate token found in the dictionary then no need to asses features, simply assign a tag. Second if a word fails to fulfill the condition of the dictionary each and every feature of the candidate token is taken to compute the averaged weight of the label. Then Training the model performs in three phases as stated in the learning the weight.

- initialize the weight and threshold values to a small random value in between (0-1)
- For each example j in the training data set D we perform the following action on the bases of the input xj and desired output dj

  $y_i(t)=f[w(t).x_j] = f[w_0x_{j,0} +w_1(t)x_{j,1}+\ldots+w(t)x_{j,n}]$

- Update the weight

  $W_i(t+1)=w_i(t)+(d_j-y_j(t))x_{j,i}$ for all $0<=i<=n$

- Step two repeated until the error is less than the specified iteration rate.

  Error rate can be specified as

Conceptually, we're looking at a whole big set of training data. We take this data one view at a time, and for each observation, update weights one at a time. The variables can be described as follows:

i: the index of the weight we're currently updating.
$w_i(t+1)$: the next condition of the i^{th} weight.
$w_i(t)$: the current condition of the i^{th} weight.
d_j: the j^{th} observation. It's represented by either 1or 0.
y_j: the j^{th} predicted observation. It's also represented by either 1 or 0.
$x_{i,j}$: the data input for the j^{th} observation.

4.3.5 Feature Extraction

The default algorithm uses different features that can be indicator of best tag sequence of currently active token. Features are used to improve accuracy of the model. The feature used in the algorithm is the main mechanisms that are expected to improve the model performance. Basically the default feature set for the default model is 13 sets of features and all features are utilized with some modification to fit Tigrigna features.

1. the suffix of the current word (i suffix)
2. the prefix of the current word (i prefix)
3. Previous tag (i-1 tag)
4. Previous two tag (i-2 tag)
5. Combination of the two previous tags i tag+i-2 tag
6. The word at position i with the tag at position i-1 (i-1 tag + i word)
7. The word at position i-1(i-1 word)
8. The word at position i-1 suffix (i-1 suffix)
9. The word at position i-2 (i-2 word)
10. The following word at position i+1 (i+1 word)
11. The word at position i+1 suffix (i+1suffix)
12. The word at position i+2 context of word at position i+1 based on their prefix
13. Following word at position i+2 (i+2 word)

Note that, there are two aspects of features choices; First, it is necessary to choose the types of features to be used as templates to generate features of training data. Then a set of features are filtered according to a number of occurrences in training dataset. Usually it is not suitable to use all features generated from the training dataset, only those appeared more than the specified threshold is required. This is substantially decreasing the iteration number and makes the algorithm ran faster. It was experimentally found that the minimum number of occurrences should be 3, because it does not decrease accuracy and the time effect is significant (Matthew, 2013). In general averaged perceptron tagger takes un-tagged Tigrigna text and produces labeled dataset as final output. So, to create a common understanding with rule based tagger and averaged perceptron tagger the word tag separator is set to backslash (/) and no space is found among the tag and the word as in the example.

ኮፍ/V መበሊ/V ውን/Av እምንን/PN ኣእዋምን/PN ኢዩ/V ።/Pu ኣንታ/Dt ወይናይ/Aj ንኡስየ/Nውፈርሞ/V ዳሕን/V ተመለስ/V እናተብሃለ/PV መርዓዊ/Nኣዕርክቱን/PN ወረዲ/Aj መርዓን/PN ሒዙ/V ናብ/Pp እንደጓል/PN ድሕሪ/Pp ሰዓት/N ይኸይድ/V።/Pu ወራዲመርዓ/Cp እንዳጓል/PN እናደረፉ/PV የኣትዉ/V ።/Pu ጓሶት/N ድማ/Cj ፀኒሖም/V ሆ/Ij !/Pu እናበሉ/PV የኣትዉ/V።/Pu ፀወታ/N ጠጠው/V ክብል/V ተገይሩ/V ብወገን/PAj እንዳጓል/PN ንእንዳወዲ/PN ዳይና/N ’ዶ/Cnኣለኩም/V ?/Pu ይብሃሉ/V ።/Pu

4.4 Hybrid Tagger

The biggest weakness for a TEL tagger is time needed for training phase since it uses large templates. Especially, it took several minutes when it combined with statistical models. TEL tagger by itself results from a very large, normally overlapping and sometimes “incorrect" set of rules, that's why we are inspired to combine the result of tagger for better performance. on the other hand averaged perceptron tagger uses averaged weighted values to guess a sequence of suggested tag for every candidate token however, there is no maximum or minimum score limit that can be used as a convergence value of the tagger . But it takes relatively small scale time to train for significantly large amount of dataset.

Specifically, we can explain the biggest strength of a Brill tagger as the model makes sense; in particular it is easy to store rules for a human-readable format as compared to statistical models.

Manually examining statistical tagger is so tedious, error-prone and not very useful, on the other hand a set of transformation rules can be understood and interpreted manually, but this can be done even by people with no previous experience in NLP (Gebregzabiher, 2010; Mekuria, 2013). Sample rules are stipulated in appendix C.

In short, Brill tagger is useful as the last step in a robust tagger for error detection and correction (Asress, 2008; Gebregzabiher, 2010; Mekuria, 2013). Instead using Brill tagger alone it is better to use Brill tagger on top of another tagger, preferably a combined system such as voting or threshold value examination. When you setup three or four different taggers, use a voting function to select the best tag for each token and only then feed these results to a Brill tagger. Hopefully, Brill tagger corrects the most common mistakes of the previous tagger and the process is tested and reported on different language including locally (Asress, 2008; Gebregzabiher, 2010; Mekuria, 2013).

Threshold values analyzer module is another way of integrating rule based approach to other approach as in the work of (Asress, 2008; Gebregzabiher, 2010; Mekuria, 2013). The main reason that we are inspired to combine the tagger is to eliminate drawback of individual tagger, considering improving to perform accuracy.

One tagger can make more tagging errors than another tagger in one field but gives better results in another field. Results presented by Asress (2008), Gebregzabiher (2010), and Mekuria (2013) shows combination of taggers score higher performance than individual taggers. There are several approaches to decide between tags suggested by taggers. In our experiments we uses fixed threshold value to combine averaged perceptron tagger with rule based tagger. The designed model and how we implemented is present as follow:

As explained averaged perceptron tagger takes unseen Tigrigna text data. Then all raw text datasets is passing through preprocessing component of the model. Afterward, the averaged perceptron tagger is initiated to assign tag to every candidate token based on features set function and dictionary collection. Note that, in training the averaged perceptron tagger the maximum weight holder suggested tag is returned to a candidate token. For this work purpose we limit the averaged weight returned for all features of 100% to create common understanding and to make it

convenient for the hybrid model (i.e. we create averaged weighted value normalization process). Thus, the maximum weighs returned to final tag is accessed and attached to the candidate token. Afterward, a tag was assigned to a candidate token until the length of the sentence is reaching. Then, all averaged weighted score values assigned to each member of a sentence was summed and divided by length of the sentence. Then, we set fixed threshold values that help the model to decide whether the tags assigned to sentence member were taken as a final tag or the sentence requires further assessment. Accordingly, if the final summed weighted score satisfies the fixed threshold value the model considers the tags as a final tag otherwise the entire sentence passes to Brill tagger for further assessment. The preprocessing stage was done only in the first step and both the sentencetokenizer(), wordtokenizer() and normalization process was done at the very beginning of the model. For Brill tagger all the combined tagger was used as initial state tagger. Finally, output of the model was compared against the reference data for performance measure.

In general the hybrid tagger performs as two step processes of four components. The main components of the tagger are preprocessor component, trained averaged perceptron tagger, output analyzer, and trained rule based tagger. Initially, Tigrigna texts are feed to trained averaged perceptron after all texts are normalized and processed in preprocessing component. Then average perceptron component assigns tag for each candidate based on the specified features and the concept of the word/tag occurrence. Then output analyzer takes advantage of fixed threshold value expression to analyze and validate output of averaged perceptron tagger. The output analyzer has two components. First the max score value returned by the prediction function of averaged perceptron tagger to each tag for each candidate token which are members of a sentence was summed and divided by the total length of the sentence as in the work of (Mekuria , 2013) .

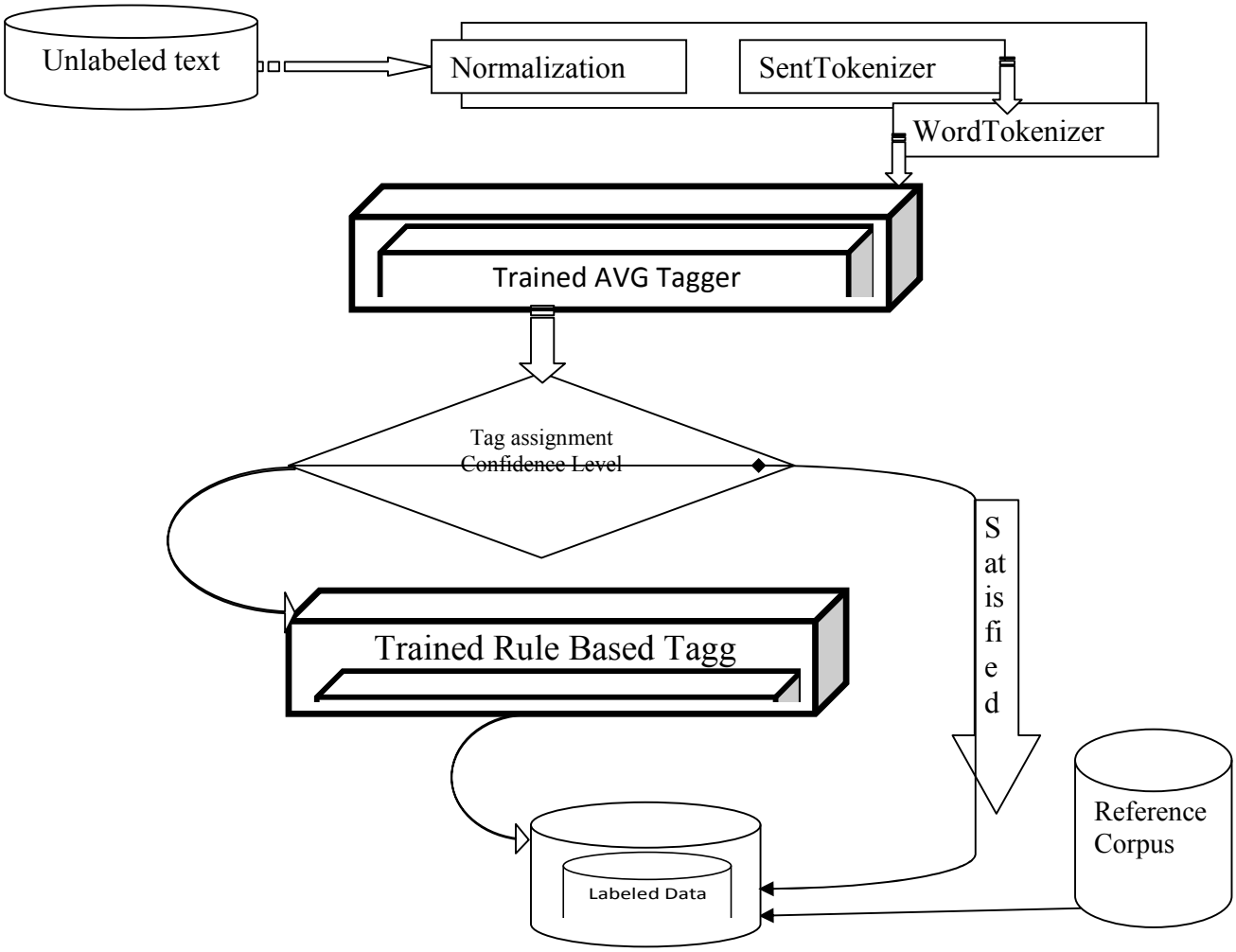


Figure 4.5: Hybrid Tagger for Tigrigna Language

$$\frac{\sum_{i=0}^{len(sen)} wi*xi}{len(sen)}$$

Figure 4.6: Mathematically expression of confidence level computation (Mekuria, 2013)

Second, the threshold value level expression was triggered to examine whether the final result score was greater than the fixed confidence level or not. In the second main operation, the output analyzer decides whether or not the confidence level of the entire sentence that is obtained from the first main operation of the output analyzer is greater than or equal to that of the fixed thre-

shold value. The threshold value is a predefined value used to check a confidence level of tagging a given sentence. If the summed score value of the entire sentence member is greater than or equal to that of the predetermined threshold value, the sequence of tags that are assigned to the entire sentence does not need any further assessment. Otherwise, the entire sentence passed to next component of the tagger (rule-based component). The general architecture and technical expression of the hybrid model was present in Figure 4.6 and Figure 4.7 respectively.

1. Input unseen Tigrigna Text
2. Set experimental threshold value
3. Get trained Averaged Perceptron tagger
4. Get trained Brill tagger
5. Tag input text using Averaged perceptron tagger
6. Get score values of each tag given the word while Averaged Perceptron tagger assigning tag for the word
7. While there are Averaged Perceptron tagged sentence
   Add the score values of each word within the sentence and divide with the number of token within the sentence length
   If the score of the entire sentence > = threshold value then
   The sentence tagged with the trained Averaged Perceptron tagger considered to be the correct one (output)
   Else
   Apply Brill tagger to the entire sentence
   The sentence that is tagged by Trained Brill tagger considered to be the correct one (output)
8. End of Averaged Perceptron tagged sentence

Figure 4.7: Technical Expression of Algorithm for the hybrid tagger (Mekuria, 2013)

CHAPTER FIVE

EXPERIMENTS AND ANALYSIS

5.1Overview

Among the NLP tasks Tagging is a process used as input for the other tasks and information retrieval to develop stemming algorithm. Tigrigna tagger was implemented for corpus from different genres using Natural Language Toolkit and Python. Different experiment has been conducted and detail explanation of each experiment is provided with the next sections.

5.2 Corpus Preparation

Corpus refers to sets with or without additional information. Corpus collection can contain text, audio, multimedia collection and so on (Tedla, Kazuhide, & Ashuboda, 2016). To make the corpora more useful for doing linguistic and language technology research, they are often subjected to a process known as tagging. Corpus with additional linguistic information in the form of word class information is known as tagged (annotated) corpus (Tedla, Kazuhide, & Ashuboda, 2016). Basically, for this work the term 'corpus' mean to clearly indicate that Tigrigna text collection (TGC) in the form of word/tag sequence.

5.2.1 Corpora Collection

Currently compilation of raw text corpora is no longer a big problem, because most documents are written in machine readable format and are available on the web. Collecting raw text corpora is a little more difficult problem with Tigrigna language as compared to English even locally as Afan Oromo and Amharic due to Tigrigna documents available in the web are limited in number. Therefore, to the best of the researcher collect, process and annotate a corpus for this work is the main constraint and big challenge that we experience while doing this study. After a long effort some Tigrigna raw text was collected, processed and used as a corpus for all experiments conducted in this study.

As explained in chapter 1 section 4.2 Tigrigna raw texts were collected from different sources. Statistical property of the corpus was presented in chapter 1 section 4.2 Table 1. Even size of the corpus compared to other language is not as large enough; however, it is possible to say sufficient for this work and they are not domain specific. But, the corpus is 3 times larger than the previous work done for Tigrigna.

As explained earlier because text document was collected from different sources most texts are unorganized and unprocessed. Thus, to make it more convenient for tagging, and to process easily the collected raw text are subjected to preprocessing component as follows:

5.2.2 Corpus Preprocessing

Pre-processing refers a process of cleaning raw text corpora for further assessment (Tedla, Kazuhide, & Ashuboda, 2016). It also refers to a process of preparing raw text corpora to make it convenient for a given model especially for language processing task like NLP application and information retrieval. Thus, raw dataset need to be cleaned and processed before feeding into automatic annotator (Mekuria, 2013). Therefore, using the cleaned dataset we can produce permissible and acceptable result particularly they have significant impact for tagger performance (Gebregzabiher, 2010; Mekuria, 2013). The process can be done either manually or automatically. Many programming languages are available to perform the activity automatically; for this study python programming language is used. As a result, a number of programs were developed to process and normalize the collected text.

Manual annotating of unorganized and unprocessed data may be easy to human language experts. However, it needs careful attention, time, effort and it is also tedious. Feeding unorganized and unprocessed data onto automatic text annotator results language property errors and affect performance of the tagger. Even it can cause fatal error that totally produces misconception to machine annotator (Yoshimasa et al, 2005). The collected raw text contains sentences with typographic errors, sentence and words with-out delimiters, unfinished sentences, normalized text etc. Moreover, the aforementioned limitation is processed and cleaned for the stated misconception and error result. The process is presented as follows:-

First the entire corpus was converted to the form of '.Txt ' that was convenient to process by python models. After all lines of no value (Null values) were discarded, sentence level arrangement was made (sentence per line). Then values contain one or more than one sentence in a single sentence was eliminated (sentence without delimiters). i.e. For this reason as explained earlier the main problem of tagging at sentences level , was wrongly tagging at the beginning of a sentence may cause wrong tag assignment to the entire sentence , because , this can create a big ambiguity during automatic annotation . Thus, we eliminate it as a best solution to the corpus for this work purpose only. We believe that elimination of such sentences will help the tagger to understand contextual information about word sequence in a sentence. Afterward, sentence members were split into token level to split into token with no space among and to identify word sequence. On the other hand, accessing raw text at word level helps us to perform normalization in an easy manner. See the following and below examples; it contains two words (ትዕዝብቲዘይምክያዱ). Each word 'ትዕዝብቲ'(tOzbti) 'ዘይምክያዱ' (zeymkyadu) has its own meaning. However, they appear as one word in the original corpus.

ኣብቲኣብ እቲ

ኣብኢ ኣብ እዚ

ከምኡ'ዩ ከምኡ እዩ

'ሎ ኣሎ

'ዩእዩ

'ዮምእዮም

ትዕዝብቲ፣ዘይምክያዱ 'ትዕዝብቲ' '፣' 'ዘይምክያዱ'

መዝጊብካምሓዝ(Record – Keeping) 'መዝጊብካ' 'ምሓዝ'

ትዕዝብቲ:ዘይምክያዱ 'ትዕዝብቲ' ':' 'ዘይምክያዱ'

In all cases words attached together with special character and punctuation marks were selected from the file and edited automatically using python regular expression tokenizer and checked

manually for the availability of the correct punctuation mark in between as follow. ኣመጋግባ ። (ቅድሚ 'ኣመጋግባ': '(': 'ቅድሚ'

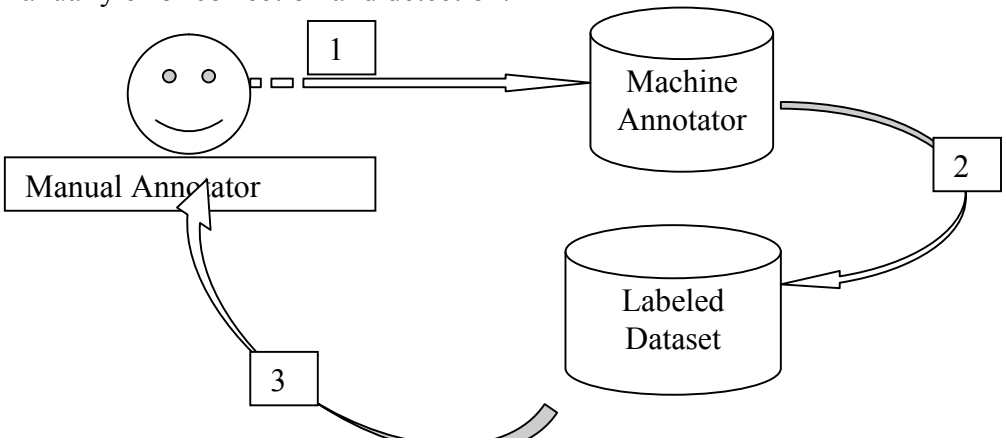
The above word contains two separate words; but the two words are attached together to open parentheses "("so it should be separate to give the correct meaning. Whereas the following two words should no separate because they lose their meaning and should read as a single word together in order to maintain their meaning in the sentence. (ኢድን-ጋንትን).

Finally sentence per-line arrangement was done and the corpus was formatted to '.Txt ' format and used as corpus in this study. As to the researcher this text file is clean and ready to be used for other Tigrigna research without any amendment especially for speech synthesis tasks. In the next section detail explanation of corpus annotation and validation was present. Sample unprocessed Tigrigna texts were provided with appendix A.

5.2.3 Annotated Corpus Preparation

To prepare annotated corpus we follow an incremental (semi-supervised) annotated corpus preparation approach (Gebregzabiher, 2010), (Mekuria, 2013) and (Addisu, 2016). This is due to resources limitation and requires time and effort of linguistic expert. The processing strategy can be defined in three phases. Figure 5.1 shows annotated corpus preparation stages:

1. Initial state manually annotated data
2. Unseen data was tagged using the machine annotator,
3. Manually error correction and detection.



5.1: Annotated Corpus Preparation Process (Mekuria, 2013)

The process starts with manually annotated first 100 sentences because there is no readymade labeled data to start over. Using the first 100 sentences another 200 sentences were tagged automatically. Then, labeled outputs were given to language expert for error correction and validation. Again the corrected 200 sentences were added to the first 100 sentence and using 300 sentences another 300 new untagged sentences were tagged automatically. Then step number 3 and 1 repeated. This process was preformed 10* times until the required annotated corpus size was retained. After a long process the required size of corpus was computed. Afterward, the annotated corpus was used to trained and test the model performance. Sample Tigrigna corpus with flat and tagged version is shown in Appendix B.

5.3Test Results

Several experiments have been conducted for Tigrigna PoS tagger. To do these, the entire corpus was divided into training and testing set considering the size of corpus in hand (Gebregzabiher, 2010; Mekuria, 2013). The training set covers 75% of the entire corpus and the remaining 25% of the corpus is used for testing purpose.

5.3.1 Test Result of Averaged Perceptron Tagger

To see the goodness of the training set, we conduct 20 different experiments with different portion of the training dataset using NLTK based Averaged perceptron tagger. Because, NLTK module provides an excellent corpus reader, first the corpus was read in sentence level with Tigrigna sentences delimiters (!?።፧፨), and then each word tag sequence was used as tuple of (word/tag) sequence in a sentence as separated by white space delimiter. Afterward, training model object was constructed with a little modification to fit Tigrigna.

To conduct the experiments, first the researchers divide the entire training dataset into 20 partitions to see models performance with small scale training dataset. Then, the researchers train the model using 5% of the training set. After training, performance of the trained model is measured

using to test dataset. Then, the process is repeated by adding 5% to the preceding portion in fact, the desired performance of the model is considered to be the performance measured from the learning curve when all the training set (100%) is consumed. Table 5.2 and Figure 5.2 show experimental result with original averaged perceptron tagger. The original averaged perceptron tagger performance varies as training dataset increase. This is due to the feature formats and arrangement of templates.

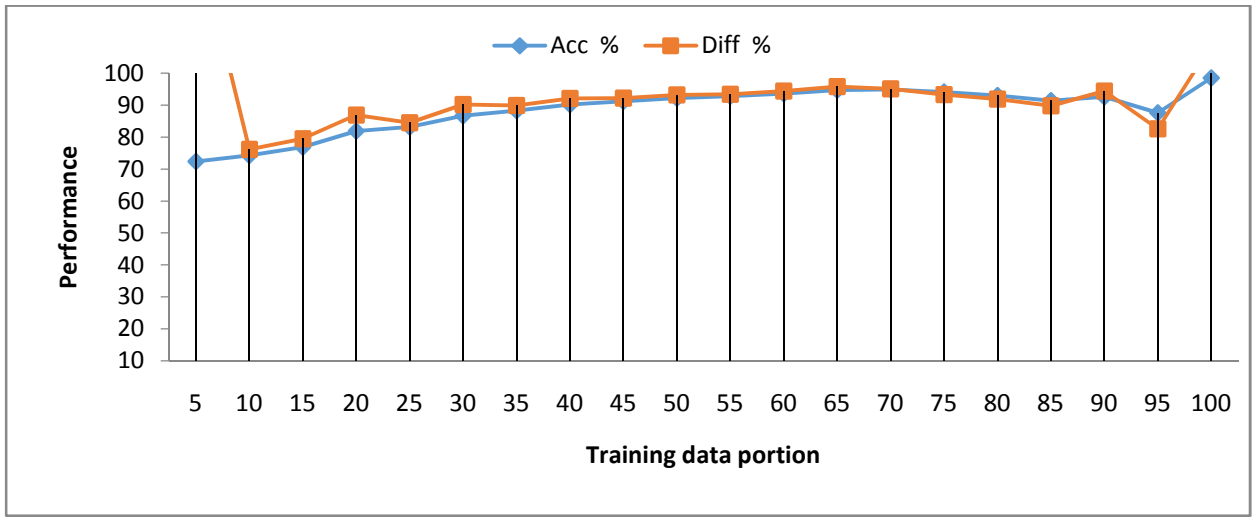


Figure 5.2: Performance curve analysis of OAVGT with different training dataset

Table 5.1 and Figure 5.3 shows experimental result of MAVGT after ambiguity threshold values set to 99%, frequent words of collection dictionary to 20% and iteration range in between 1-10. The tagger Performance is also increased as the iteration values increase and training dataset increases. This indicates, as iteration value increases possibility of finding the true tag increases. However, the tagger produces stable result as the iteration values rich in 6 and above and taking iteration value less than 6 affects performance of the tagger.

Table 5.1: MAVGPT Performance Curve Analysis with different iteration values

Iteration rang	1	2	3	4	5	6	7	8	9	10
Accuracy %	92.3	93.2	93.6	94.1	94.3	95.5	95.5	95.5	95.5	95.5
Difference %	92.3	0.9	0.4	0.5	0.2	1.2	0	0	0	0

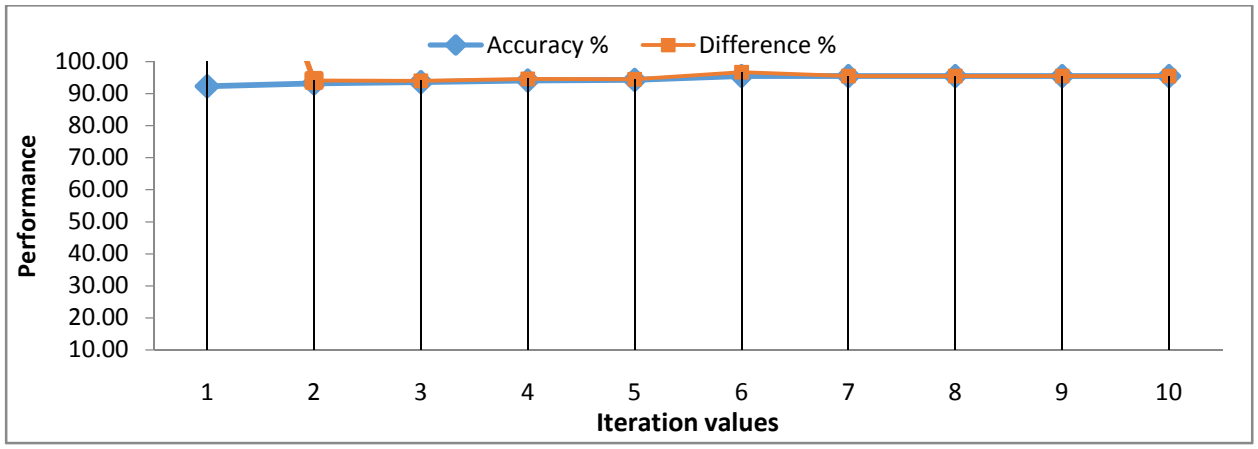


Figure 5.3: Performance curve Analysis of MAVGPT with different Iteration values

Therefore, the tagger performance is good when all parameters are mounted to fit Tigrigna features; like adding Tigrigna punctuation mark identifier function, frequent word counter dictionary set to 20% and ambiguity threshold value set to 99, Iteration value set to 6, and training dataset increase accordingly . Since Tigrigna language uses a set of different punctuation marks of along to the common punctuation mark used for other languages it is important to introduce the Tigrigna punctuation with the tagger. So the function created to introduce Tigrigna punctuation mark has significant impact on the tagger performance. Similarly the frequent words with the dictionary are set to 20% and the ambiguity threshold values were set to 99% due to the nature and statically property of the corpus. Afterward, another experiment has conducted to see the model performs with the aforementioned training dataset scales. The main reason for iteration value is 6, taking >6 the model produces stable result. Moreover, it was expected to reduce iteration loop and learning curve time of the tagger. Table 5.3 and Figure 5.4 shows performance curve analysis MAVGT with different portion of the training dataset. Thus, performance of the tagger is increased to 95.5%. This indicates us, if the features are extended to deep feature set function then; possibility of identifying true tag increase. So, we can consider the key parameters that lead the tagger to better performance is feature function, the frequent word dictionary ambiguity threshold values and the frequency of word counter. The training iteration value of the taggers is also a key parameters used to increase performance of the tagger.

Table 5.2: Performance Curve Analysis of OAVG PT

TRP	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100
Acc	72.4	74.3	76.9	81.9	83.2	86.7	88.3	90.2	91.2	92.2	92.8	93.6	94.7	94.9	94.1	93	91.4	92.6	91.6	91.6
Diff	72.4	1.9	2.6	5	1.3	3.5	1.6	1.9	1	1	0.6	0.8	1.1	0.2	-0.8	-1.1	-1.6	1.8	-5	10.4

Table 5.3: Performance Curve Analyses of MAVGPT

TRP	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100
Acc	78.1	83.2	85.2	88.6	91.1	92.3	92.55	93.3	94.07	94.4	94.75	95.1	95.2	95.24	95.3	95.23	95.29	95.33	95.41	95.50
Diff	78.1	5.1	2	3.4	2.5	1.2	0.25	0.75	0.77	0.33	0.35	0.35	0.1	0.04	0.06	-0.07	0.06	0.04	0.08	0.09

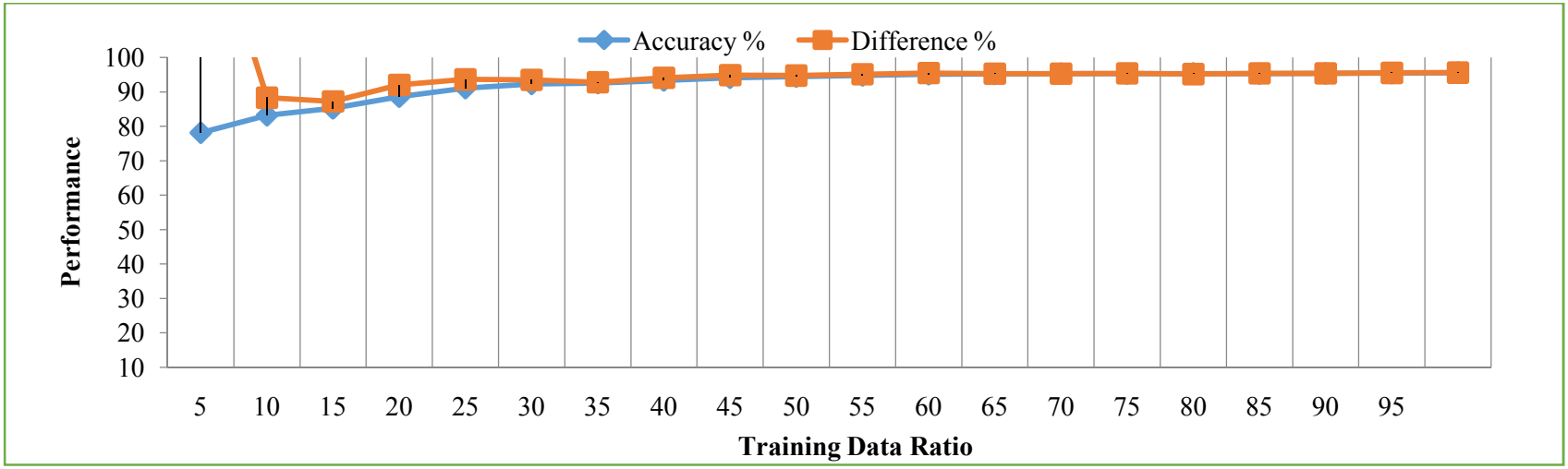


Figure 5.4: Performance curve analysis of MAVGPT with different training dataset

5.3.2 Test Result of Rule-based Tagger

Different experiments have been conducted with rule based tagger along three combined initial state annotators. Two different templates were developed based on the default templates. Thus, including the original template the experimental result has reported as follows:-

Experiment A

As explained in chapter 4 section 4.2 Brill tagger has initial state annotator that can range from simple to complex initial state annotator. Thus, the first sequence tagger was set to Regular expression tagger and its main task is to assign tag to candidate tokens based on provided patterns. It also assigns a label 'Rs ' as a Default tag for words has no match in the patterns. According to frequency of individual tag identified with the entire corpus and nature of Tigrigna, Noun (N) tags are identified as a default word class. However, as clearly explained the drawback of default tagger in chapter 4 section 4.2 we use 'Rs ' tag label to handle unknown word as a default tag. To develop the patterns some specific property of each class was identified with the help of language expert and Tigrigna grammar book references. Then, Affix Tagger learns prefix and suffix of a candidate tokens. In this stage morphological property of candidate token is used to determine best tag for the candidate token. Thus, tag is assigning based on affixes and suffix attached to root word. Based on morphological property of Tigrigna words and corpus collection property in hand we set a prefix length to '4 ' and a suffix length to '-4'. In addition to this, the minimum lengths of candidate token were set to '2 ' to handle all words that have length of '2'. The reason for taking the minimum length word character is the corpora collection in hand. For instance most preposition in Tigrigna has length of '2 ' characters. Again all tokens passed through affix tagger were feed to unigram tagger. The unigram tagger assigns tag for each word based on word/tag frequency in training corpus; i.e. it selects the most likely tag for a given word

Table 5.4 and Figure 5.5 show different experiment conducted using different portion of training dataset with corresponding performance of rule-based tagger along the stated initial state annotators, with original Template that contains 24 templates.

Table 5.4: Performance Curve Analysis of Regular Expression, Affix and Unigram Tagger

TRP%	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100
RegT	35.60	35.20	37.8	37.8	37.8	37.8	37.8	37.8	37.8	37.8	37.8	37.8	37.8	37.8	37.8	37.8	37.8	37.8	37.8	35.7
AfixT	40.3	41.0	45.2	47.3	48.9	49.2	49.3	49.	49.5	49.5	49.5	49.7	49.7	49.8	49.8	49.7	49.7	49.7	49.9	50.5
UgT	63.8	67.9	73.6	78.6	81.6	82.6	83.0	84.2	85.3	85.9	86.5	87.0	87.1	87.4	87.6	87.7	87.9	88.2	89.0	89.2
BrT	**66.8**	**71.2**	**74.8**	**79.6**	**83.3**	**83.8**	**85.3**	**85.3**	**86.1**	**86.4**	**87.3**	**87.7**	**87.9**	**88.2**	**88.2**	**88.4**	**88.7**	**88.9**	**89.7**	**90.4**
Dif f%	66.80	3.60	4.80	4.80	3.70	0.50	1.5	0.0	0.8	0.3	0.9	0.4	0.2	0.3	0.0	0.2	0.3	0.2	0.8	0.70

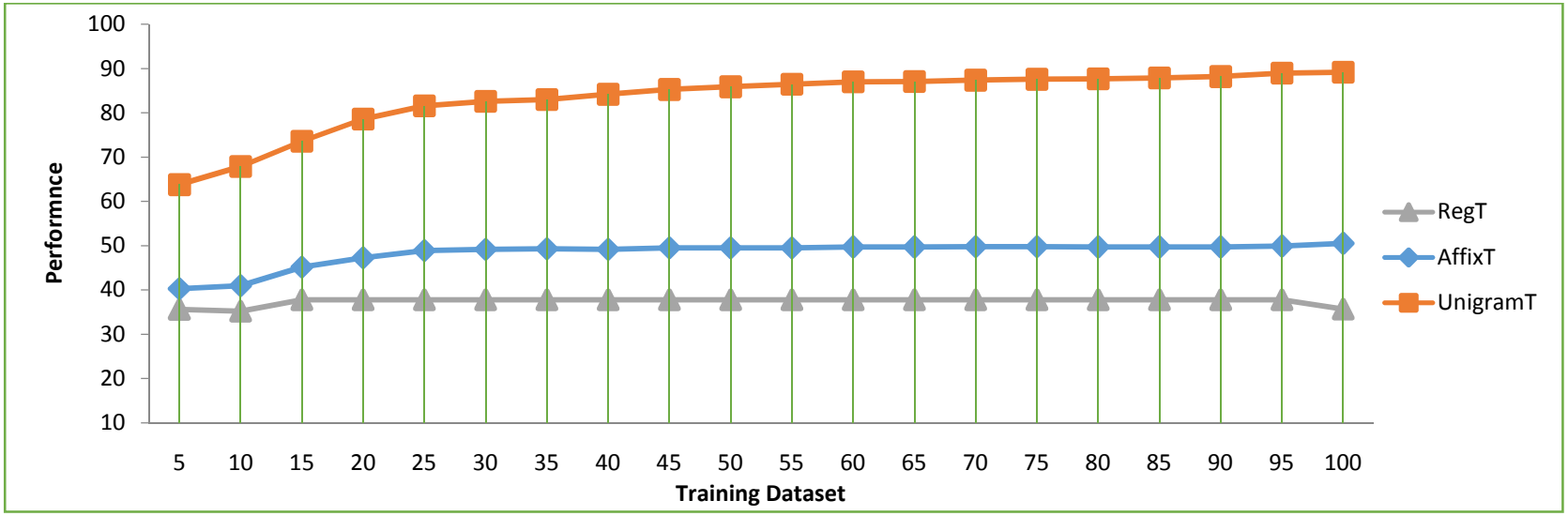


Figure 5.5: Performance Curve AnalysisOf Initial State Annotator With Different Training Dataset

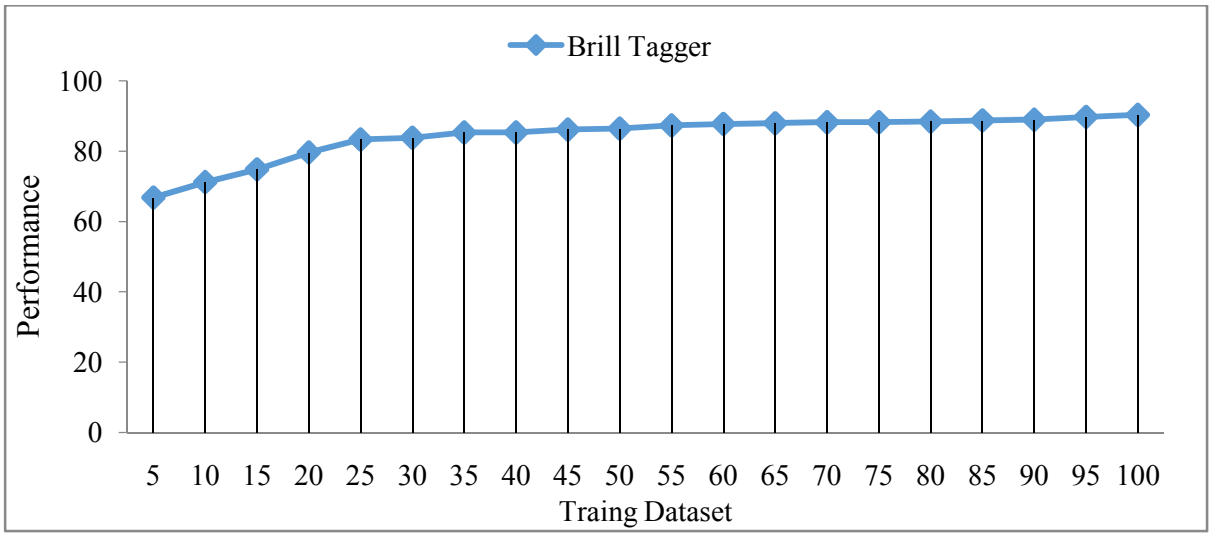


Figure 5.6: Performance Curve Analysis of Rule Based with Original Template

Experiment B:

During this experiment the default template size was expanded upon 28. The templates are also different from their arrangement and forms into the default one to fit Tigrigna. That means we made some modification in the original template arrangement and expand their size by 4 and all combined initial state annotator are used as in the first experiment. Table 5.5 and Figure 5.7 shows performance curve analysis of the conducted experiment with the modified 28 templates. As explained due to word variation distribution, nature of Tigrigna sentence and morphological behavior of Tigrigna, performance of the tagger was affected. However, it performs better than the original tagger. The extended template includes:-

1. Word at position W(0) and PoS at Position PoS(-1)
2. Word at position W(0) and PoS at Position PoS(1)
3. Word at position W(0), Word at position W(3) and PoS at Position PoS(3).
4. Word at position W(0), Word at position W(2) and PoS at Position PoS(2). This enables us to use advantage of context of a word at positional three and its tag.

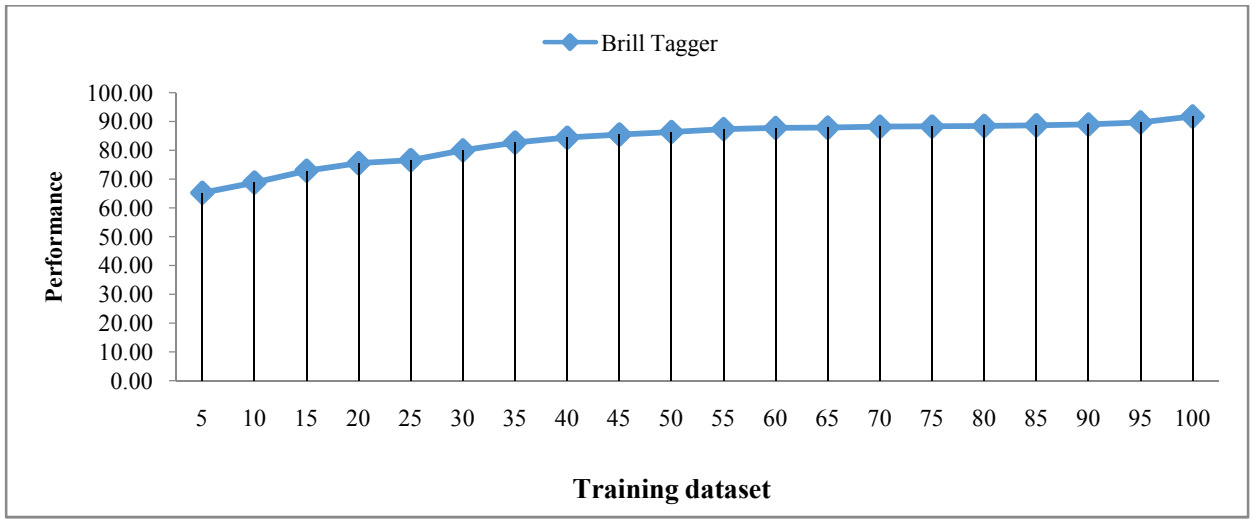


Figure 5.7: Performance Curve Analysis of Rule Based tagger with 28 Templates

Again we reduce size of our modified template for 23 and the templates are different from their arrangement and size. Mainly, the reduced templates are more related to contextual rule so as to see the effects of the contextual templates in relation the model. So we reduce the modified template by 5.Accordingly, the reduced template affects the performance of the tagger. Therefore, contextual information collected during learning phase is a promising direction to the tagger performance from the predefined contextual templates. Table 5.6 and Figure 5.8 show experimental result of Brill tagger with 23 templates with different portions of the training dataset.

Table 5.5: Performance Curve Analysis Of Rule Based Tagger With 28 Templates

TRP	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100
BrT	65.2	68.8	72.9	75.5	76.6	80.0	82.7	84.4	85.5	86.4	87.4	87.8	87.9	88.2	88.3	88.5	88.7	89.0	89.7	94.8
Dif	65.2	3.60	4.10	2.60	1.10	3.40	2.70	1.70	1.10	0.90	1.00	0.40	0.10	0.30	0.10	0.20	0.20	0.30	0.70	2.10

Table 5.6: Performance Curve Analysis Of Rule Based Tagger With 23 Template

TRP	5	10	15	20	25	30	35	40	45	50	55	60	65	70	75	80	85	90	95	100
BrT	67.5	71.5	74.7	79.6	83.3	83.9	84.1	85.3	86.3	86.7	87.4	87.9	88.0	88.2	88.4	88.5	88.7	89.0	89.7	90.4
Dif	65.5	4.00	3.20	4.90	3.70	0.60	0.20	1.20	1.00	0.40	0.70	0.50	0.10	0.20	0.20	0.10	0.20	0.30	0.70	0.70

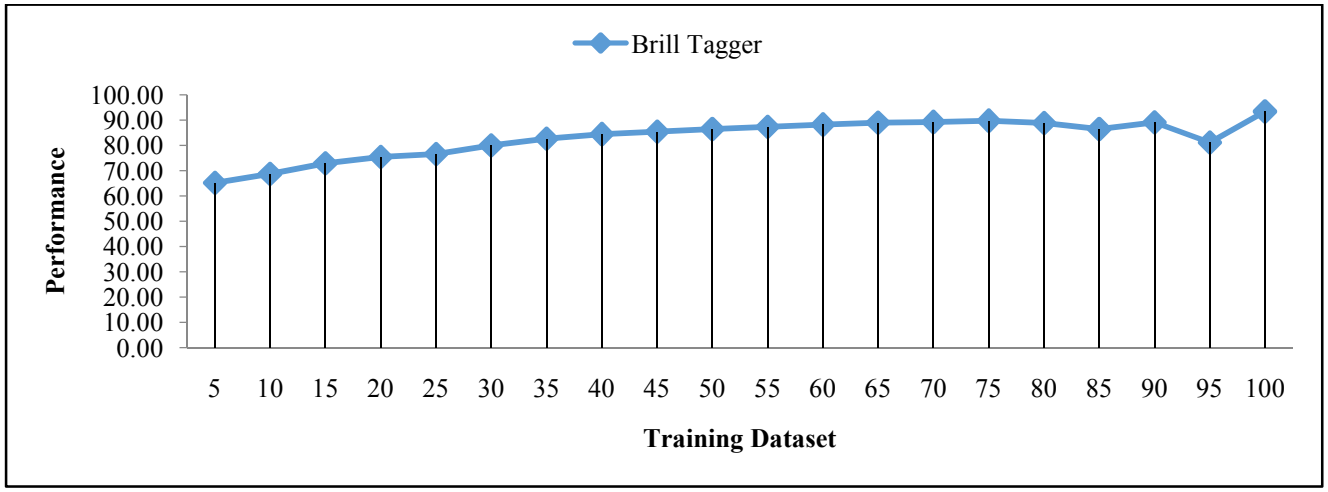
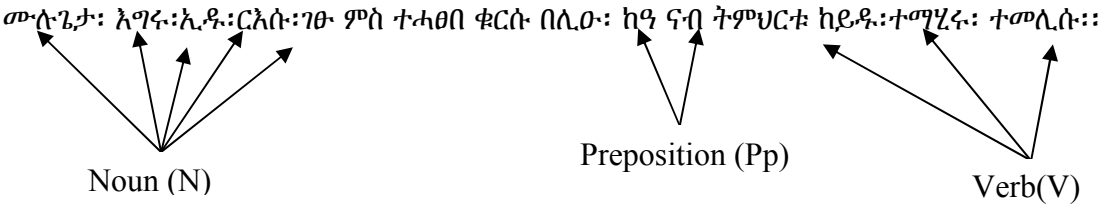


Figure 5.8: Performance Curve Analysis of Rule Based Tagger with 23 Templates

Therefore, there is no big difference between the experimental result for the original rule based tagger and rule based tagger with modified template 23. However, the tagger performs well with 28 templates and increases it performance. Thus, the contextual information about candidate token is a promising direction for improving the tagger performance. The main reason for decreasing performance of the tagger are Tigrigna sentences complexity and sequence of words that have the same class categories appearing sequentially takes significance portion. Mainly, the word difference can be attached as affixes and clearly show in their actions on the serious. For instance, Lets look at the sentence and word sequences.



From the above sentence ሙሉጌታ (mulugieta)፡እግሩ (Agru)፡ኢዱ (idu)፡ርእሱ (rAsu)፡ገፁ (geTu) are noun appear in the sentence sequentially . From this concept, assume that our target word is ኢዱ (hand): according to our template arrangement we assess the two consecutive words and two consecutive tags that follow and preceded the target word. Accordingly, the two preceding words with the target word are ሙሉጌታ (mulugieta) and እግሩ (Agru) and the two following words are ርእሱ (rAsu) and ገፁ (GeTu) . On the other hand the tow preceded tags are "N ' and 'N ' as well as the two following tags are 'N' and 'N'. According to the word sequence and behavior of the sentence it is difficult for automatic annotator to generalize the word class of the current word as to the manually annotator. Therefore, the possibility of ambiguity was high. Another thing was the purpose of word with the sentence. For instance the word (መሐሪ) (meHari) can be noun if it is a person's name and it can be verb in the case of doing an action. Finally from all the experimental result reported, the modified 28 template was taken as the final template to error rate analyses and to construct the hybrid model.

5.3.3 Test Result of Hybrid Tagger

The hybrid taggers were constructed from sequence of two taggers. The first component is made from averaged perceptron tagger and the second component is made from rule based tagger. All components of averaged perceptron and rule based tagger were tuned to the best of the researcher knowledge to fit Tigrigna. Initially averaged perceptron tagger was used to annotate unseen word sequence. Accordingly, if the sum result of averaged score of individual tag divided by the length of the entire sentence is less than the expected fixed confidence level, the entire sentence pass to rule based tagger for further correction. However, if the computed weighted value of the entire sentence fit the expected fixed confidence level, then the assigned tag is considered as a final label. Considering this, different experiment was conducted with different threshold values. To do this, first all as-signed maximum weighted values is normalized to 100 percent. The normalization processes helps to fix the value of less than 1. Then all maximum values returned to sentence member candidate token is retrieved; Then summations of all values is computed. Afterward, the sum values of a sentences member are divided by sentence length. Then, the computed value is examined against the fixed confidence level value. Since the maxim range is in between (0 and 1) we take as a best level in range 0.46 to train and evaluate the hybrid tagger. Behind taking threshold value confidence level less than 0.46 does not bring significant difference between the performances of the tagger. However, as the threshold value increase, the possibility of the correct tag for a candidate token and performance of the tagger is also increase. By fixing the confidence level to 0.46, an overall performance of 96.3% is obtained. But when the threshold value is <0.46 the performance goes down this can be as a result of sentence length variation and most of the sentences are expected to contain less number of words. Thus, most of the sentences are expected from the bible sources. Experimentally, from the corpus an average of '6' words are found in a sentence. Table 5.7 and Figure 5.9 show performance of hybrid tagger with different threshold value.

Table 5.7: Performance Curve Analysis of Hybrid Tagger with Different Threshold Value

Threshold value	0.02	0.16	0.26	0.36	0.46	0.56	0.66	0.86	1
Performance (%)	94.1	94.15	94.21	95.54	96.3	96.1	96.1	96.1	90.31

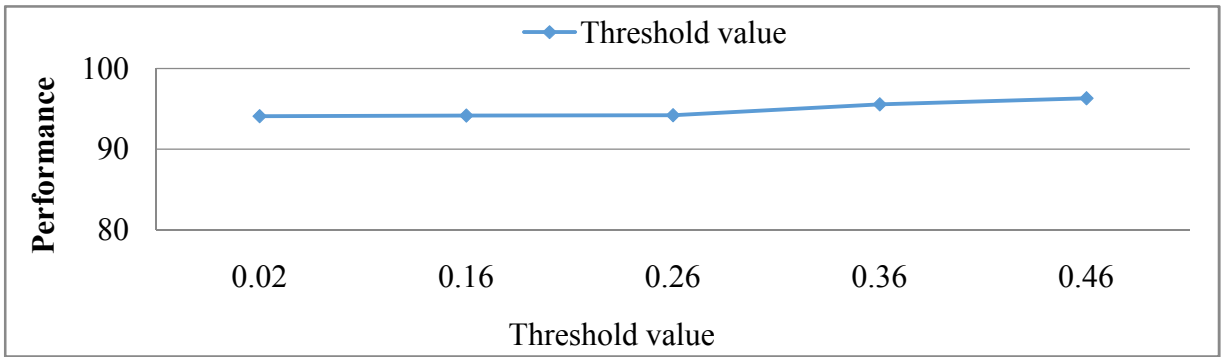


Figure 5.9: Performance Curve Analysis Of Hybrid Tagger With Different Threshold Values

5.4Tigrigna PoS Tagger Performance Analysis

To analyze performance of the tagger we develop a confusion matrix considering distribution of tags within the entire corpus, training corpus and gold standard dataset. A confusion matrix is a table used to analyses performance of a model in general PoS in particular on a set of test data for which the true values are known. Confusion matrix helps us to asses in which tag the model was most confused and in which tags the model was performing well. Since some tags are rarely occurring in the corpus we clustered the tags into one as others (Mekuria, 2013) and (Addisu, 2016). On the other hand, error committed by the tagger was also computed using simple program.

5.4.1 Tag-Set Distribution in the Entire Corpus

To analyze performance of designed models in relation to tag-set distribution tag-set frequency is considered. Afterward, confusion matrix is used to compute goodness of the taggers (averaged perceptron, rule-based and hybrid taggers) perform on each tag. The entire tags are divided in to two parts considering their frequency, first the researcher identify 8 most frequent tags and the rest to others as in the work of Mekuria (2013) and Addisu (2016). Table 5.8 shows tag-set frequency distribution of the entire corpus.

Table 5.8: Tag-set Frequency Distribution in the Entire Corpus

Tag Name	Total Tag-set Frequency	Tag-set Frequency distribution in percentage
N	20820	33.60%
Aj	3640	5.87%
V	11948	19.28%
Pn	3075	4.96%
Av	1083	1.75%
Dt	2374	3.83%
Cp	213	0.34%
Ij	22	0.04%
Cn	0	0%
Ab	117	0.19%
Nu	2567	4.14%
PN	3316	5.35%
PV	2393	3.86%
PAj	506	0.82%
PCj	1	0.00%
PNu	314	0.51%
PPp	60	0.10%
PPn	86	0.14%
Cj	1461	2.36%
Pu	3075	4.95%
Fn	5	0.01%
Pp	4885	7.88%
PN	3316	5.35%

Table 5.9: Tag-set Frequency Distribution in Different Dataset

Tag-Name	Entire corpus tag(Freq)		Training dataset tag(Freq)		Test dataset Tag(Freq)	
	Total count	In %	Total count	In %	Total count	In %
N	20820	33.60%	15590	34.13%	5369	36.29%
Aj	3640	5.87%	2951	6.46%	1118	6.50%
V	11948	19.28%	9460	20.71%	2832	18.89%
PV	2393	3.86%	1840	4.03%	559	3.71%
PN	3316	5.35%	2619	5.73%	860	5.65%
Pu	3075	4.95%	2521	5.52%	672	4.51%
Pp	4885	7.88%	3702	8.11%	1314	8.82%
Nu	2567	4.14%	1840	4.03%	768	4.95%
Others	6878	15.07%	9880	0.1228	1733	10.68%
Total	60893	100.00%	50403	100.00%	15225	100.00%

5.4.2 Confusion Matrix of Averaged Perceptron Tagger

As we can see from Table 5.10 averaged perceptron tagger confusion matrix assigns 16848 tags correctly out of 17543 tags. Just like that of the rule based tagger the performance of the model was varying from each tag. Accordingly, it shows higher performance for Pu, Pp, N, Nu, V, PN, and PV, Others, followed by Aj.

Table 5.10: Confusion Matrix of Averaged Perceptron Tagger

		Predicted tags(test dataset)										
		N	V	Pp	Aj	PN	Nu	Pu	PV	Others	Total	Percentage
Actua tags (reference datsset)	N	36.6%	0.4		0.2	0.1					6451	97.83%
	V	0.5	18%								3293	95.93%
	Pp			8.7%							1545	99.09%
	Aj	0.3	0.3		5.8%	0.1					1150	87.99%
	PN	0.1				5.2%					947	94.03%
	Nu						4.6%				831	97.59%
	Pu							4.5%			795	100.%
	PV					0.1			3.5%		661	92.73%
	Others		0.1			0.2				9.8%	1870	92.63%

1) (6, ((('ዞባ', 'N'), ('ሰሜናዊ', 'Aj')), (('ዞባ', 'N'), ('ሰሜናዊ', 'N'))))
2) (5, ((('ንምክያድ', 'PV'), ('ምድላዋት', 'V')), (('ንምክያድ', 'PV'), ('ምድላዋት', 'N'))))
3) (4, ((('ገበረ', 'V'), ('ኣወዳትን', 'PN')), (('ገበረ', 'V'), ('ኣወዳትን', 'N'))))
4) (4, ((('ከቢድ', 'Aj'), ('ስጉምቲ', 'V')), (('ከቢድ', 'Aj'), ('ስጉምቲ', 'N'))))
5) (3, ((('ድሕሪኡ', 'Pp'), ('ሾሞንተ', 'N')), (('ድሕሪኡ', 'Pp'), ('ሾሞንተ', 'PNu'))))
6) (3, ((('ዘሎ', 'V'), ('ልምዓት', 'N')), (('ዘሎ', 'V'), ('ልምዓት', 'V'))))
7) (3, ((('ውሽጣውን', 'Aj'), ('ዓለምለኻውን', 'N')), (('ውሽጣውን', 'Aj'), ('ዓለምለኻውን', 'Cp'))))

From the error analysis in the first case the model assigns 'Aj' tag 6 times incorrectly for the word ሰሜናዊ (semeinawi) in the test dataset which is 'N' in training dataset. The same is true to the second analysis the model assigns 'V' tag to the word ምድላዋት 5 times incorrectly in the test data-set. However, the correct tag is 'N' in training dataset. The same is true to the rest of the error output.

5.4.3 Matrix of Rule Based Tagger

The rule-based tagger confusion matrix assigns 16623 tags correctly out of 17543 tags. Performance of the tagger varies from each tag. Accordingly, it shows higher performance for Pu, Pp,

N, Nu, Other, PV, Aj, V, followed by PN. See the confusion extracted from the model during tag assignment.

Table 5.11: Confusion Matrix of Rule Based Tagger

		Predicted tags(test dataset)										
		N	**V**	**Pp**	**Aj**	**PN**	**Nu**	**Pu**	**PV**	**Others**	**Total**	**Percentage**
Actual tags (reference data-sset	**N**	36.2%	0.2		0.1	0.1					6451	98.50%
	V	1.5	17.1%			0.1			0.1		3293	83.37%
	Pp	0.0		8.7%							1545	98.64%
	Aj	0.4	0.2		5.8%						1150	83.64%
	PN	0.6				4.7%					947	80.50%
	Nu	0.1	0.1				4.5%				831	90.01%
	Pu							4.6%			795	100.00%
	PV	0.2	0.1		0.1				3.4%		661	85.91%
	Others	0.3	0.1						0.1	9.7%	1870	89.66%

1. (3, ((('ኣብ', 'Pp'), ('ዝተፈላለዩ', 'Aj')), (('ኣብ', 'Pp'), ('ዝተፈላለዩ', 'V'))))
2. (2, ((('ፕሮሞሽንን', 'PN'), ('ውለቀ', 'N')), (('ፕሮሞሽንን', 'PN'), ('ውለቀ', 'Aj'))))
3. (2, ((('ፕሬዝደንትን', 'PN'), ('ዝመረፀ', 'N')), (('ፕሬዝደንትን', 'PN'), ('ዝመረፀ', 'V'))))
4. (2, ((('ፕላንን', 'PN'), ('ሰነድን', 'N')), (('ፕላንን', 'PN'), ('ሰነድን', 'PN'))))
5. (2, ((('ፎረም', 'N'), ('ክሳተፉ', 'N')), (('ፎረም', 'N'), ('ክሳተፉ', 'V'))))

From the error analysis in the first case the model assign ‘V’ tag 3 times incorrectly for the word ‘ዝተፈላለዩ’ in the test dataset which is ‘Aj’ in the training dataset. The same is true for the second analysis the model assign ‘N’ tag to word ‘ውለቀ’ 2 times incorrectly in the test dataset. However, the correct tag in the training dataset is ‘Aj’ the same is true for the rest error analysis.

5.3.5 Confusion Matrix of Hybrid Tagger

The confusion matrix curve analysis of the hybrid tagger shows the tagger assigns 545 tags incorrectly out of 17543. The main source of the confusion were identified as the nature of the lan-

guage features , appearance of the same word class level sequentially and incorrectly labeling of tags in the reference corpus . A give word assigned two or more than two tags available in training dataset based on the role of the word with the sentence. However, as we can see from Table 29 the Committed by the tagger are less than errors produced by the individual taggers. Considering the actual tag and predicted tag performance of the hybrid tagger varies from one tag to another tag. Table 5.12 shows performance variation on the tagger for each tag. Thus, the tagger performs well on tag Pu followed by PP, V, Nu, and PN, N, Aj, PV and others. Because, we integrate a function that simply introduce Tigrigna punctuation mark to the tagger, its performance is perfect with all punctuation mark. The tag 'N' confuses vertically and horizontally with tag 'Aj' and the tag 'PN' and tag 'V' this confusion could be from the word sequence appearance in the sentence, miss tag assignment in the reference corpus and training corpus. Again the tag 'N' is most confused with the tag 'Aj' this could raise because of the functionality of the words in the context.

Table5.12: Confusion Matrix of Hybrid Tagger

		Predicted tags(test dataset)										
		N	V	Pp	Aj	PN	Nu	Pu	PV	Others	Total	Percentage
Actua tags (reference datsset	N	36.6%	0.4		0.2	0.1					6451	97.83%
	V	0.3	18.2%								3293	96.05%
	Pp			8.7%							1545	100.00%
	Aj	0.3	0.2		5.9%	0.1					1150	88.34%
	PN	0.1				5.2%					947	94.44%
	Nu						4.6%				831	97.95%
	Pu							4.5%			795	100.00%
	PV		0.1			0.1			3.5%		661	92.12%
	Others		0.1			0.2				9.8%	1870	92.31%

Some of the errors commuted by the hybrid tagger were listed as follow. The main causes of the confusion are may be from the nature and complexity of the Tigrigna. I.e. word sequence, typographic error or more than one tag bound with single word in the test dataset.

1. (4, ((('ከም', 'Pp'), ('ዝኾነ', 'V')), (('ከም', 'Pp'), ('ዝኾነ', 'PV'))))
2. (3, ((('ኣብ', 'Pp'), ('መንነ', 'Aj')), (('ኣብ', 'Pp'), ('መንነ', 'N'))))
3. (3, ((('ሚኒስተር', 'N'), ('ጉዳያት', 'Aj')), (('ሚኒስተር', 'N'), ('ጉዳያት', 'N'))))
4. (2, ((('ፍርያት', 'N'), ('ክሰጋገሩ', 'V')), (('ፍርያት', 'N'), ('ክሰጋገሩ', 'PV'))))

Error analysis in the first case the model assigns 'V' tag 4 times incorrectly for the word ዝኾነ (zkone) in the test dataset which is 'PV' in the training dataset. The same is true for the fourth analysis the model assigns 'V' tag to the word ክሰጋገሩ 2 times incorrectly in the test dataset. However, the correct tag is 'PV' in the training dataset. The same is true for the rest of the error output.

Discussion of results

Different experiments have been conducted for Tigrigna averaged perceptron tagger, Rule based tagger and Hybrid tagger (a sequence of averaged perceptron and rule based tagger). The taggers were trained and tested for the prepared annotated corpus. Accordingly, performance of the Original averaged perceptron Tagger and Modified averaged perceptron tagger is measured to 91.6% and 95.5% respectively, with the difference of 4.5%. Performance of the modified averaged perceptron Tagger was better than the original averaged perceptron Tagger. The reason for performance of the MAVGT increases could be from the morphological features of the language integrated into the feature function, the iteration values, the ambiguity threshold values for frequent word dictionary collection or it could be the collection corpus help the tagger to train for more diverse sentences.

Performance of original rule based Tagger and modified rule based tagger along the integrated initial state annotator is measured to 90.4% and 94.8% respectively, with difference of 3.9%. The main reason behind increasing performance of the tagger could be the different sources that integrate with the rule based tagger beside the contextual and lexical information. Afterward, a combined sequence of averaged perceptron tagger followed by rule based tagger as error detection and correction model was designed. Like the individual taggers the hybrid model were also trained and tested for the aforementioned datasets. Because the hybrid model combines advantage of the two individual taggers it achieves a reasonable tagger for Tigrigna language. Thus,

performance of the hybrid tagger was measured for 96.3%. The number of error committed by the tagger was less than error committed by individual taggers. The hybrid tagger reduce error rate by 1.5% from rule based tagger and it reduces error committed by averaged perceptron tagger by 0.8% in average the hybrid tagger reduce error rate for 1.15%.

Therefore, all Tigrigna taggers achieve state-of-the-art of tagging for morphological rich languages. The main reasons for performance improving on the taggers are, the researcher prefers to include morphological property of the language instead of using statistical information. On top of that, we integrate different concepts that clearly identify word class information on the tagger like patter, affixes and statistical information of the candidate tokens. The statistical property integrated with the tagger help not ignore information about rarely tokens from the corpus. As this work is a second work for the language, size of the corpus is three times greater than the corpus of the previous work and is not domain specific corpus. Because, the corpus collection contains text from different sources all taggers are well trained unlike that of the previous work done which was trained and tested for a single source. The researcher believes that from rule based tagger a set of permissible rules are generated with the corpus. So, it is possible to conclude new dataset doesn't affect performance of the tagger. However, to some extent performance of the tagger could be degrade but it doesn't affect the entire performance ;i.e. as the number of unique words with the test dataset increases accuracy of the tagger is not affected. With the averaged perceptron tagger the feature functions are the main components of the tagger that can be extended to deep feature sets that have a promising direction for performance improvement. The feature function can be extended for features like gender, number, and tens and can be also includes tag-set designed at low level with deep morphological features of the language. The designed hybrid tagger is also well trained and uses the advantage of the two taggers; first it eliminates drawback of averaged perceptron tagger of convergence value even if the convergence value was set at sentence level; second it uses the grammatical property of rule based tagger as error detection and correction sequence. Again the fixed threshold value integrated into the output analyzer is also another important factor that shows great impact on the performance of the hybrid tagger. Because all taggers are trained and tested for the corpora not domain specific, performances of the taggers are high. Generally, the performance score measured for each tagger and their difference is presented within Table 5.13.

Even though, the researcher found reasonable PoS taggers for Tigrigna, the performance of the tagger could be enhanced if the tagger was trained for large size of annotated corpus.

Table 5.13: Summary Score Performance of Tigrigna PoS Taggers

Tag Name	Avg Tagger in %	Rule Based Tagger in %	Hybrid Tagger in %	Difference between rule based & Avg tagger in %	Difference between AVG & Hybrid in %
N	97.83%	98.50%	97.83%	0.67%	0.00%
V	95.93%	83.37%	96.05%	-12.56%	0.12%
Pp	99.09%	98.64%	100.00%	-0.45%	0.91%
Aj	87.99%	83.64%	88.34%	-4.35%	0.35%
PN	94.03%	80.50%	94.44%	-13.53%	0.41%
Nu	97.59%	90.01%	97.95%	-7.58%	0.36%
Pu	100.00%	100.00%	100.00%	0.00%	0.00%
PV	92.73%	85.91%	92.12%	-6.82%	-0.61%
Others	92.63%	89.66%	92.31%	-2.97%	-0.32%

CHATPTER SIX

CONCLUSION AND RECOMMENDATION

6.1 Conclusion

Tagging is a process of associate word class markers for corpora contents either manually or automatically. For several years several, several approaches have been proposed to the problems of tagging for different languages internationally and locally. Even though, there are many different models for tagging they are differing with their internal model or the amount of training or processing of information they need. Although there are many models and implementations available for the task of tagging, most of them are designed for and tested for English texts; less work has been done on tagging and tagger evaluation for languages like Tigrigna.

The task of PoS tagging is not trivial because many words are ambiguous. Thus, considering its importance it remains as a hot research area for other morphologically rich language in general specifically for Tigrigna language. Among the available PoS models stochastic based approach and rule-based approach are the most widely used one. The stochastic based PoS tagger uses simple and complex statically expression to perform the process. The probability is a fundamental building block of the stochastic based PoS taggers. On the other hand rule based PoS tagger uses a set of rules to perform the process. The rules are the fundamental property of rule based approaches.

Now a day's a hybrid of two or more approaches were proposed to different languages internationally and locally. The main goal of combining two or more taggers is to improve tagging accuracy. Tagger Sequence order and voting are the most wildly used way of combining tagger. Processed and organized corpora with or without additional information is important for many language technology researches including tagging. Thus, the study uses a corpus containing 3100 Tigrigna sentences collected from different genres that are expected to represent morphology and nature of Tigrigna.

The collected Tigrigna texts are subjected to preprocessing activity before used as a corpus for this study. Afterward, using semi-supervised annotated corpus preparation method the promised corpus was computed.

For this studya total of 22 Morpho-Syntactic tag-set at course-grained level is used to annotate Tigrigna corpus. Since the tag-set are labeled with course-grained level they only provide general word class category information on every member of the corpora rather than gender, number, tenses, infix etc.

This paper describes experiments conducted with Rule based to approach, Averaged Perceptron based and Hybrid approaches for PoS tagging of Tigrigna text. A corpus with additional corpora content of the form of word class markers was created for these experiments and the taggers were trained for 75% of the corpus and were tested for the remaining 25%. The first experiment was conducted with the default and modified Averaged Perceptron tagger, These Averaged Perceptron taggers were based on modified feature function and ambiguity threshold values, training iteration. The feature function includes Tigrigna word morphology and context of words with in a given sentence based on word orders sequence. Thus, with modified Averaged Perceptron tagger we achieve a reasonable PoS tagger that increases performance accuracy as the training data-set size increase.

Our Averaged Perceptron tagger results show that these feature functions has significant impact on improving the tagging accuracy of the model.

The second experiment with the Default Rule based tagger we generate a reasonable PoS tagger for Tigrigna. However its performance is low due to the nature and morphological behavior of the language. Afterward, a bit modification was made to default Brill tagger and on the integrated initial state annotator considering performance of the tagger. Then, we also experimented with the modified Brill tagger. Based on our experimental report we achieve PoS tagging state-of- the-art performance accuracy for morphological complex and rich languages specifically Tigrigna language with modified 28 templates Rule based tagger along with three different initial state annotators.

The third experiment was conducted with the hybrid tagger considering accuracy performance of the tagger and performance of the two individual taggers. According to our experimental report as we expected the hybrid approach shows better performance than individual tagger performing alone. Since we mount both tagger to the best of the researchers know-ledge and skill to fit Tigrigna language property we increase performance accuracy of the tagger to 96.3%. This is because to the tagger uses advantages of grammatically induced rules from rule based tagger with sentences that are failed to pass threshold value expression and the feature function included in the averaged perceptron tagger.

The overall performance of Rule based approach to three different initial state annotators and the Averaged perceptron tagger is comparable. However, the hybrid approach increases performance of the tagger. According to our experimental report 95.5%, 94.8% and 96.3% accuracy performance was obtained for Averaged Perceptron, rule-based and hybrid taggers respectively. From our experimental report it is possible to conclude that tagging Tigrigna sentences based on hybrid approach from sentence level to perform better than using the individual taggers alone.

Finally, from our experimental analysis we observe that lack of standard testing data-set, complexity of morphology and ambiguity of Tigrigna word, lack of documented references, scarcity of available resource and lack of integrating deep features of Tigrigna language to the tagger are the most potential factors that confuse the tagger to commit errors.

6.2 Recommendation

Associating additional word class information about corpora contents of the form of word class marker is a useful task in both linguistic and language technology field. Besides attaching additional information of corpora content, the task can serve as an earlier step component for many others higher levels NLP takes information retrieval tasks. Therefore, researchers involving in the field of NLP and Information retrieval can use the designed model as preprocessing component within their research. However, nature and morphological complexity of Tigrigna the tagger face lack of integrated sophisticated features at low level including fine-grained and very-grained tag-set. Even we develop a corpus collected from different sources but the information added to the corpus is general information which doesn't include low level features like gender, tens and number.

There are different approaches designed to solve the problem of tagging internationally and locally, thus, this work can be used as a basement to assess different approaches on top of the work done for Tigrigna language. Because, we observe that the designed tagger performance is better than the previous work done for Tigrigna individually and in the hybrid taggers. Thus, as a future work, we would like to suggest the following key points:

- Extending this work using Morpho-Syntactic tag-set at fine-grained and very-fine-grained level that can identify deep features of Tigrigna
- Comparative study using same tag-set, training and testing dataset using different approaches
- Investigating, test and train other approaches for Tigrigna
- Improve performance of individual taggers by extending features of Tigrigna.
- Testing and evaluating the tagger for other language

References

Achmid, H. (1994). Part-of-Speech Tagging With Neural Networks. *Institute for Computational Linguistics*

Addisu, B. (2016). Development Of Part Of Speech Tagger For Sidaama Language. Master's Thesis, Addis Ababa University.

Ager, S. (2016). *The Online Encyclopidia of writting system and language* . Retrieved May 14, 2016, from Tigrinya (ትግርኛ): http://www.omniglot.com/writing/tigrinya.htm

Asress, S. (2008). Automatic Amharic Part-of-Speech Tagging Using Hybrid Approach (Neural Network and Rule-Based). Unpublished Master's thesis, Addis Ababa.

Bird, S., Klein, E., & Loper, E. (2005). *NLTK Tutorial: Introduction to Natural Language Processing.* Stanford, California, USA.

Brants, T. (2000). A Statistical Part-of-Speech Tagger. *In Proceedings of the Sixth Applied Natural Language Processing Conference ANLP.* Seattle, WA: Saarland University.

Bril, E. (1992). A simple rule-based part of speech tagger. *In Proceedings of the Third ACL Applied NLP*, (pp. 152-155). Trento.

Brill, E. (1995). Transformation-Based Error-Driven Learning and Natural Language ProcessingA Case Study in Part-of-Speech Tagging. *Association for Computational Linguistics.* Johns Hopkins University.

Christopher, M. (2000). *Foundations of Statistical Natural Language Processing.* Massachusetts: MIT Press Cambridge.

Collins, M. (2002). *Discriminative Training Methods for Hidden Markov Models:*. New Jersey: AT&T Labs-Research.

Daelemans, W., Zavrel, J., Berck, P., & Gillis, S. (1996). A Memory-Based Part of Speech Tagger-Generator. *In proceedings WVLC, Computational Linguistics and AI* . Copenhagen: Tilburg University.

Das, D., & Petrov, S. (2011). Unsupervised Part-of-Speech Tagging with Bilingual Graph-Based. *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, (pp. 600–609). Portland.

Gasser, M. (2011). HornMorpho: A System For Morphological Processing Of Amharic, Oromo, And Tigrinya. *Conference on Human Language Technology for Development.* Alexandria, Egypt.

Gebregzabiher, T. (2010). Part Of Speech Tagger For Tigrigna Language. Unpublished Master's thesis, Addis Ababa University.

Hafte, A. (2009). Hidden Markov Model Based Tigrigna Speech Recognition. Unpublished Master's Thesis, Addis Ababa University.

Hasan, F. M., UzZaman, N., & Khan, M. (n.d.). Comparison of different POS Tagging Techniques (N-Gram, HMM and Brill's tagger) for Bangla. *Center for Research on Bangla Language Processing.* Bangladesh .

James & Daniel. (2006). *Speech and Language Processing. An introduction to natural Language Processing,*. New Jersey.

KHOJA, S. (2014). APT: Arabic Part-of-speech Tagger. Lancaster: Lancaster University.

Kriesel, D. (2005). *A Brief Introduction to Neural Networks.* Germany.

Loftsson, H., & Östling, R. (2013). Tagging A Morphologically Complex Language Using An Averaged Perceptron Tagger The Case of Icelandic. *Proceedings of the 19th Nordic Conference of Computational Linguistics.* Linköping Electronic Conference Proceedings.

Matthew, H. (2013, September 18). *A Good Part-Of-Speech Tagger In About 200 Lines Of Python*. Retrieved Augest 7, 2016, from A Good Part-Of-Speech Tagger In About 200 Lines Of Python: https://explosion.ai/blog/part-of-speech-pos-tagger-in-python

Mekuria, Z. (2013). Design And Development Of Part-Of-Speech Tagger For Kafi-Noonoo Language. Published, Master's thesis, Addis Ababa University.

Michael & Nigel,. (2002). *NewRankingAlgorithmsforParsingandTagging:*. NewJersey: AT&TLabs-Research.

Mirzanur, R., Sufal, D,. & Utpal, S. (2009). Parsing Of Part-Of-Speech Tagged Assamese Texts. *IJCSI International Journal of Computer Science Issues, Vol. 6, No. 1* .

MUAIDI, H. (2014). Levenberg-Marquardt Learning Neural Network For Part-Of-Speech Tagging Of Arabic Sentences. *WSEAS TRANSACTIONS on COMPUTERS* .

Nazareth, A. (2011). Tigrinya Applicative in Lexical-Functional Gramma. Doctoral dissertation, University of Bergen.

Nedjo, A. T., & Liu, D. H. (2014). Automatic Part-of-speech Tagging for Oromo Languagesing Maximum Entropy Markov Model. *Journal of Information & Computational Science*,11-10.

NLTK Project. NLTK 3.0 documentation. (2015, April 09). Retrieved December 12, 2016, from NLTK Project. NLTK 3.0 documentation.: http://www.nltk.org/py- modindex.html

Okbeab, T. (2014). Phonetic-Based Keyboard Mapping For Tigrigna Language. Master thesis, Strathmore University.

Parikh, P. (2009). Part-Of-Speech Tagging using Neural network. *7th International Conference on Natural Language Processing.* Macmillan: India.

Python Software Foundation. (2016). *Python Software Foundation*. Retrieved July 21, 2016, from State-Of-The-Art Part-Of-Speech Tagging In Textblob: https://pypi.python.org/pypi/textblob-aptagger

Rowl, R. (1996). *The Neural Networks Complexity of Learning.* Berlin.

Sahilu, A. (1998). *Sewasw Tigrinya Bisefiu- A comprehensive Tigrignya grammar.* Asmara : Red see Press.

Tateishi., et al. (2005). Developing A Robust Part-Of-Speech Tagger For Biomedical Text., (pp. 382–392). Springer-Verlag, Berlin, Heidelberg.

Tedla, et al. (2016). Nagaoka Tigrinya Corpus: Design and Development of Part-of-speech Tagged Corpus. *The Association for Natural Language Processing.* Nagaoka University,Japan.

Teffera, T. (1979). *Reference Grammar Of Tigrigna.* Wahsington, Dc, USA: Georgetown University.

Teklu, D. (2008). *Zemenawi Sewasiw Tigrigna language.* Adiss Ababa.

Tewelde, T. (2002). *A Modern Grammar Of Tigrigna.* Savonarola, Roma: Tipografia U. Detti – via G.

Tsegay, W.(2014). A Morphsyntactic Tag-set For The Annotation Of Tigrigna Corpora, (TIG-TAGS). Zena lissan, Vol. XXIII, No2, pp(99-112). Addis Ababa University.

Zhang, Y., & Clark, S. (2010). Syntactic Processing Using the Generalized Perceptron and Beam Search. *Association for Computational Linguistics* .

Appendices

Appendix A: SampleUnprocessed Corpus Version

ዜና ድሕሪ ቀትሪ *ዕለት21/08/2007*

ኣብ ወረዳ ደጉዓ ተምቤን ሃገረሰላም ሙሉእ ቀዳማይ ብርኪ ቤት ትምህርቲ ንህንፀት ኣደራሽ ሰረት እምነ ኩርናዕ ተነቢሩ።

ልምዓት ዓዲ ብወዲ ዓዲ ብዝብል ኣምር ተወላዲት ወረዳ ደጉዓ ተምቤን ነባሪት ስዑድ ዓረብ ዝኾና ወይዘሮ ኣልጋነሽ ኣብርሃ ኣብ ሙሉእ ቀዳማይ ብርኪ ቤት ትምህርቲ ሃገረሰላም ፀገም መኣከቢ ኣዳራሽ ንምቕራፍ ብሰናይ ተበግሶ ንምስራሕ እምነ ሰረት ተቀሚጡ።

እተን ግዱስ ዜጋ ተምሃሮ ኣብ ዘፈር ትምህርቶም ንፍዓት ኾይኖም ክወፁን ኣሰር ኣያታቶም ስዒቦም ኣብ ልምዓት እዛ ሃገር ክነጥፍ ዝካኣለን ንምግባር ድሉውነተን ዘረጋገፃ እንትኾና እቲ ኣደራሽ ድማ ብሸም ስውእ ተጋዳላይ ዮሃንስ ኣብርሃ ተሰይሙ እንትህነፅ ንብዙሓት ተወለድቲ ኣዛ ከባቢ ዘለዓዕል እዩ ኢለን ኣለዋ።

ሰራዊት ልምዓት ትምህርቲ ፈጠርቲ ፣ፍቐሪ ህፃን፣ ሃገርን ሕ/ሰብን ዘለዎም/ን ኩለ መዳያዊ ስብእና መምህርነት ዝተላበሱን/ሳን ንድሑር እምነታት ተሪሮም/ረን ዝቃለሱን/ሳን ንሳይንስን ቴክኖሎጅን ጠመተ ብምሃብ ብሞያ፣ ክእለት፣ ኣረኣእያን ኣካዳሚያዊ ብቕዓት ዘለዎም/ወን መምህራንን ኣመራርሓ ትምህርትን ብቐፃልነት ምፍራይ።

- ልምዓታውን ዲሞክራስያውን ኣተሓሳስባን ተግባርን ምዕባይ
- ፆታዊ፣ ብሄረሰባውን ሃይማኖታውን ማዕርነት ዘረጋገፀን ፍሉይ ድሌት ዘድልዮም ኣካላት ዘካተተን ፍትሓዊ ምልመላ ምክያድ

<u>ዝተጠቕለለ ድምር ውፅኢት</u>

i. ጉድኣት እንተጋጥም ብርኪ እቲ ጉድኣት ምምዛንን ሓገዝ ምሃብን(Evaluating and Treating Illnesses and Injury
   - ንዝለዓለ ሕክምና "ሪፈር" ምባል

ii. ድሕሪ ጉድኣት ተፃወቲ ናብ ንቡር ንኽምለሱ ምሕጋዝ (Rehabilitation)
   - ተፃወቲ ድሕሪ ጉድኣት ብቕልጡፍ ናብ ንቡር ንኽምለሱ ክሰርሕዎም ዝግብኡን ዘይግብኡን ስራሕቲ ብፅሑፍ ብምሃብ ግልጋሎት ድርዛት/ማሰጅ ምሃብ

iii. ውፅኢት መዝጊብካ ምሓዝ(Record – Keeping)
   - ዝተገበረሉ ሓገዝ ፣ ዝወሰዶ መድሓኒት ወ.ዘ.ተ ብንፁር ምምዝጋብ

ለ.ቕንዲ መንቀሊ ስፖርታዊ ጉድኣት(Causes of Sports Injuries)

- ግጭት (ምስ ሰብ፣ ሓፂንወ.ዘ.ተ)
- ናውቲ ስፖርት ብኣግባቡ ዘይምጥቓምን ንምንቅስቓስ ምችዋት ዘይምኽንን
- ዘይተስተኻኽለ (ንምንቅስቓስ ዘይምቹ) ሜዳ

Appendix B: Sample Corpus Of Untagged and Version Tagged

ኢኮኖሚዝብልደርፊህወሓትህዝቢትግራይደጋጊሙዘዝይሞምቁርደርፊእዩ።

ትግራይዓድናወርቂንኽትሽለምካብቲብላዕፅምትንደምንደቃዝተነድቐስርዓትትፅቢትእትገብሮብዙሕነገርእንተሃለወካብዚኦምእቲቀንድንወሳንንዑዞዕብየትዝኾነምስፍሕፋሕኢንሸስትመንትእዩ።

" ሃፍታምእየእንተበሉኻትኹብር፤ድኻእየእንተበሉኻትሓስር " ዝብልምስላወለድናዶይዝከረኩምኣሎ ?

እወ፤ክብሪሃገርይኹንውልቀሰባትምስዓርሰለውጢ፣ምዕባለንምህፍታምንእዩዝተኣሳሰር።

ንምህፍታምድማነቲዘላቐንዝበለፀንሃፍቲዘኻዕብት " እሴት " ዝፈጥሩመፍረይቲትካላትእንትውነንእዩ።

ስለዝኾነእውንእየእዛሃገርርእሳኣቐኒዓኣብዓለምለኻዊመድረኻትክትዛረብንዝተዛረበቶተሰማዕነትክረክብንእንተኾይኑዝዓበያሃገራትዝበፅሐኡብርኪምዕባለክትበፅሕእንተኽኢላጥራሕእዩንብልዘለና።

ሕርሻሆርቲካልቸርውሕስናምግቢካብምርግጋፅሓሊፉኣብዕብየትቁጠባሃገርእውንጥቐሙዝለዓለብምኂኑትኩረትክዋሃቦከምዝግባእሶሳይቲሳይንስካልቸርኢትዮጵያገሊፁ።

ፕሬዝዳንትእቲማሕበርንዩኒቨርስቲጅማንዶክተርፍቅሬለሜሳሕርሻሆርቲካልቸርውሕስናምግቢሃገርናብምርግጋፅዝለዓለረብሓኣ ለዎኢሎምኣለው።

ብተወሳኺእውንእቲዘፈርንውልቀሰባትኾነቁጣባሃገርንምዕባይንንሽርፊወፃኢዝለዓለእጃምስለዘለዎትኩረትተዋሂበዎክስረሐሉይ ግባእተባሂሉሎ።

ቁጣባዓዲውሽጢሃገርናኣብሕርሻዝተመሰረተእንትኮን 60 ሚኢታዊዝኸውንንፍልፍልሸርፊወፃኢዝውዕልእዩ።

ኣክስዮንማሕበርሲሚንቶናሽናልፍርያትሲሚንቶናብጎረባብቲሃገራትሰደድምግባርጀሚሩ።

እቲኣክስዮንማሕበርኣብሓደወርሒውሽጢ 800
ቶንስሚንቶንኣርባዕተሽሕንሓሙሽተሚኢትንቶንስሚንቶናብጁብቲንሱማሌላንድንሊኢኩሎ።

መጽሓፍቅዱስብሉይንሓድሽንኪዳንኦሪትዘፍጥረትኣምላኽብመጀመርታሰማይንምድርንፈጠረ።

ምድሪድማበረኻንጥራያንነበረት፡ጸልማትከኣኣብልዕሊመዓሙቐነበረ።

መንፈስኣምላኽድማኣብልዕሊማያትይዝምቢነበረ።

ኣምላኽከኣ፥ብርሃንይኹን፡በለ።

ብርሃንድማኼነ።

ኣምላኽድማእቲብርሃንጽቡቕከምዝኾነረኣየ።

ኣምላኽከኣነቲብርሃንካብጸልማትፈለዮ።

ኣምላኽነቲብርሃንመዓልቲኣውጽኣሉ።

ነቲጸልማትከኣለይቲኣውጽኣሉ።

ምሽትኩነብጊሓትውንኩነ፡ሓንቲመዓልቲ።

ኣምላኽድማ፥ንማያትካብማያትዚፈሊጠፈርኣብመንጎማያትይኹን፡በለ።

ኣምላኽነቲጠፈርገበሮ።

ነቲኣብትሕቲጠፈርዘለኦማያትድማኻብቲኣብልዕሊጠፈርዘለኦማያትፈለዮ፡ከምኡውንኩነ።

ኣምላኽከኣነቲጠፈርሰማይኣውጽኣሉ።

ምሽትኩነብጊሓትውንኩነ፡ካልኣይቲመዓልቲ።

ኣምላኽድማ፥እቲንቕጽምእንቲኺርኤስ፡እቲኣብትሕቲሰማይዘለኦማያትናብሓንቲቦታይተኣኪብ፡በለ።

ከምኡድማኾነ።

ኣምላኽከኣነቲንቕጽምድሪኣውጽኣሉ፡ነቲእኩብማያትድማባሕሪኣውጽኣሉ።

ኣምላኽከኣጽቡቕከምዝኾነረኣየ።

ኣምላኽድማ፥እታምድሪሳዕርንዘርኢዚህብብቕልን፡ዘርኡኣብርእሱዘለዎ፡ፍረከከምዓይነቱኣብምድሪዚፈሪአምተውጽእ፡በለ።

Tagged Version Of The Corpus

ኢኮኖሚ/N ዝብል/V ደርፊ/N ህወሓት/Ab ህዝቢ/N ትግራይ/N ደጋጊሙ/N ዘዝይሞ/PV ምቁር/Aj ደርፊ/N እዩ/V ።/Pu ትግራይ/N ዓድና/N ወርቂ/N ንኽትሸለም/PV ካብቲ/Dt ብኣዕፅምትን/PN ደምን/N ደቃ/N ዝተነድቐ/V ስርዓት/N ትፅቢት/N እትገብሮ/PV ብዙሕ/Aj ነገር/N እንተሃለወ/V ካብዚኦም/Dt እቲ/Dt ቀንድን/PAj ወሳንን/PAj ጉዕዞ/N ዕብየት/N ዝኾነ/V ምስፍሕፋሕ/V ኢንቨስትመንት/N እዩ/V ።/Pu ሃፍታም/N እዩ/V እንተበሉኻ/PV ትኽብር/V ፤/Pu ድኻ/N እዩ/V እንተበሉኻ/PV ትሓስር/V ዝብል/V ምስላ/PV ወለድና/N ዶ/Ij ይዝከረኩም/V ኣሎ/V ?/Pu እወ/N ፤/Pu ክብሪ/N ሃገር/N ይኹን/V ውልቀ/N ሰባት/N ምስ/Pp ዓርሰ/N ለውጢ/V ምዕባለን/PV ምህፍታምን/PV እዩ/V ዝተኣሳሰር/V ።/Pu ንምህፍታም/PV ድማ/Cj ነቲ/Dt ዘላቕን/N ዝበለፀን/V ሃፍቲ/N ዘኻዕብት/V እሴት/N ዝፈጥሩ/V መፍረይቲ/Aj ትካላት/N እንትውነን/PV እዩ/V ።/Pu ስለዝኾነ/PV እውን/Av እዩ/V እዛ/Dt ሃገር/N ርእሳ/N ኣቕኒዓ/N ኣብ/Pp ዓለም/N ለኻዊ/Aj መድረኻት/N ክትዛረብን/PV ዝተዛረበቶ/V ተሰማዕነት/N ክረክብን/PV እንተኾይኑ/PV ዝዓበያ/N ሃገራት/N ዝበፅሐኦ/V ብርኪ/N ምዕባለ/N ክትበፅሕ/V እንተኽኢላ/V ጥራሕ/Av እዩ/V ንብል/V ዘለና/V ።/Pu ሕርሻ/N ሆርቲ/N ካልቸር/N ውሕስና/N ምግቢ/N ካብ/Pp ምርግጋፅ/V ሓሊፉ/V ኣብ/Pp ዕብየት/Aj ቁጠባ/N ሃገር/N እውን/Av ጥቕሙ/N ዝለዓለ/Aj ብምኻኑ/PV ትኩረት/N ክዋሃቦ/V ከምዝግባእ/PV ሶሳይቲ/N ሳይንስ/N ካልቸር/N ኢትዮጵያ/N ገሊፁ/V ፕሬዝዳንት/N እቲ/Dt ማሕበርን/PV ዩኒቨርስቲ/N ጅማን/N ዶክተር/N ፍቅሬ/N ለሜሳ/N ሕርሻ/N ሆርቲ/N ካልቸር/N ውሕስና/N ምግቢ/N ሃገርና/N ብምርግጋፅ/PV ዝለዓለ/Aj ረብሓ/N ኣለዎ/V ኢሎም/V ኣለው/V ብተወሳኺ/Av እውን/Av እቲ/Dt ዘፈር/Aj ንውልቀ/N ሰባት/N ኾነ/N ቁጣባ/N ሃገር/N ንምዕባይን/PV ንሽርፊ/N ወፃኢ/N ዝለዓለ/Aj እጃም/N ስለዘለዎ/PV ትኩረት/N ተዋሂበዎ/V ክስረሐሉ/V ይግባእ/V ተባሂሉሎ/V ቁጣባ/N ዓዲ/N ውሽጢ/Pp ሃገርና/N ኣብ/Pp ሕርሻ/N ዝተመሰረተ/V እንትኮን/V 60/Nu ሚኢታዊ/N ዝኸውን/V ንፍልፍል/PN ሸርፊ/N ወፃኢ/N ዝውዕል/V እዩ/V ኣከስዮን/N ማሕበር/N ሲሚንቶ/N ናሽናል/N ፍርያት/N ሲሚንቶ/N ናብ/Pp ጎረባብቲ/N ሃገራት/N ሰደድ/V ምግባር/V ጀሚሩ/V እቲ/Dt ኣከስዮን/N ማሕበር/N ኣብ/Pp ሓደ/Nu ወርሒ/N ውሽጢ/Pp 800/Nu ቶን/N ሰሚንቶን/PN ኣርባዕተ/Nu ሽሕን/PNu ሓሙሽተ/Nu ሚኢትን/PNu ቶን/N ስሚንቶ/N ናብ/Pp ጁብቲን/N ሱማሌ/N ላንድን/N ሊኢኩሎ/N መጽ/N ሓፍ/N ቅ/N ዱስ/N ብሉይን/N ሓድሽን/N ኪዳን/N ኦሪት/N ዘፍጥረት/N ኣምላኽ/N ብመጀመርታ/Av ሰማይን/PN ምድርን/PN ፈጠረ/V ።/Pu ምድሪ/N ድማ/Cj በረኻን/N ጥራያን/N ነበረት/V ጸልማት/N ከኣ/Cj ኣብ/Pp ልዕሊ/Pp መዓሙቕ/N ነበረ/V ።/Pu መንፈስ/N ኣምላኽ/N ድማ/Cj ኣብ/Pp ልዕሊ/Pp ማያት/N ይዝምቢ/N ነበረ/V ።/Pu ኣምላኽ/N ከኣ/Cj ብርሃን/N ይኹን/V በለ/V ።/Pu ብርሃን/N ድማ/Cj ኹነ/V ።/Pu ኣምላኽ/N ድማ/Cj እቲ/Dt ብርሃን/N ጽቡቕ/Aj ከም/Pp ዝኾነ/V ረኣየ/V ።/Pu ኣምላኽ/N ከኣ/Cj ነቲ/Dt ብርሃን/N ካብ/Pp ጸልማት/N ፈለዮ/N ።/Pu ኣምላኽ/N ነቲ/Dt ብርሃን/N መዓልቲ/N ኣውጽኣሉ/V ።/Pu ነቲ/Dt ጸልማት/N ከኣ/Cj ለይቲ/N ኣውጽኣሉ/V ።/Pu ምሸት/N ኮነ/V ብጊሓትውን/PV ኮነ/V ሓንቲ/Nu መዓልቲ/N ።/Pu ኣምላኽ/N ድማ/Cj ንማያት/N ካብ/Pp ማያት/N ዚፈሊ/N ጠፈር/N ኣብ/Pp መንጎ/Aj ማያት/N ይኹን/V በለ/V ።/Pu ኣምላኽ/N ነቲ/Dt ጠፈርገበሮ/PV ።/Pu ነቲ/Dt ኣብ/Pp ትሕቲ/Aj ጠፈር/N ዘሎ/V ማያት/N ድማ/Cj ኻብቲ/Dt ኣብ/Pp ልዕሊ/Pp ጠፈር/N ዘሎ/V ማያት/N ፈለዮ/N ከምኡውን/Cj ኮነ/V ።/Pu ኣምላኽ/N ከኣ/Cj ነቲ/Dt ጠፈር/N ሰማይ/N ኣውጽኣሉ/V ።/Pu ምሸት/N ኮነ/V ብጊሓትውን/PV ኮነ/V ካልኣይቲ/Nu መዓልቲ/N ።/Pu ኣምላኽ/N ድማ/Cj እቲ/Dt ንቐጽ/N ምእንቲ/Av ኺርኤስ/N እቲ/Dt ኣብ/Pp ትሕቲ/Aj ሰማይ/N ዘሎ/V ማያት/N ናብ/Pp ሓንቲ/Nu ቦታይተኣኻብ/V በለ/V ።/Pu ከምኡ/Pp ድማ/Cj ኾነ/V ።/Pu ኣምላኽ/N ከኣ/Cj ነቲ/Dt ንቐጽ/N ምድሪ/N ኣውጽኣሉ/V ነቲ/Dt እኩብ/N ማያት/N ድማባሕሪ/N ኣውጽኣሉ/V ።/Pu ኣምላኽ/N ከኣ/Cj ጽቡቕ/Aj ከም/Pp ዝኾነ/V ረኣየ/V ።/Pu ኣምላኽ/N ድማ/Cj እታ/Dt ምድሪ/N ሳዕርን/N ዘርኢዚህብ/N ብቐልን/N ዘርኡ/N ኣብ/Pp ርእሱ/N ዘለዎ/V ፍረ/N ከከም/Pp ዓይነቱ/N ኣብ/Pp ምድሪ/N ዚፈሪ/N አም/N ተውጽእ/V በለ/V ።/Pu

Appendix C: Brill Tagger Learned Rules

N -> V if the Word of the following word is "በለ"

N -> V if the Word of the following word is "አሎ፦ኩ"

N -> PN if the Word of words i+1...i+2 is "ወለደ"

N -> V if the Word of words i+1...i+2 is "በላ"

N -> V if the Word of the following word is "አለው"

N -> V if the Word of words i-2...i-1 is "ሐዝካ"

Aj -> PAj if the Word of the preceding word is "አንጋረን"

PN -> N if the Word of word i+2 is "ሓባራዊ"

Sample Contextual Rules

N -> V if the Pos of the following word are "Av", and the Pos of word i+2 is "Cj"

N -> V if the Pos of the following word is "PN", and the Pos of word i+2 is "Pu"

PV -> PAj if the Pos of the preceding word is "PV", and thePos of the following word is "PNu"

PV -> V if the Pos of the preceding word is "Aj", and thePos of the following word is "Dt"

Aj -> PN if the Pos of the following word is "Av", and thePos of word i+2 is "Dt"

V -> PV if the Pos of the preceding word is "PNu", and the Pos of the following word is "Nu"

N -> PNu if the Pos of the preceding word is "Cj", and thePos of the following word is "PNu"

Appendix D: Tigrigna Language Letters and Their Translated Characters (Omniglot, 2016)

1st	order	2nd	order	3rd	order	4th	order	5th	order	6th	order	7th	order
ሀ	he	ሁ	hu	ሂ	hi	ሃ	ha	ሄ	hie	ህ	h	ሆ	ho
ለ	le	ሉ	lu	ሊ	li	ላ	La	ሌ	lie	ል	l	ሎ	lo
ሐ	He	ሑ	Hu	ሒ	Hi	ሓ	Ha	ሔ	Hie	ሕ	H	ሖ	Ho
መ	me	ሙ	mu	ሚ	mi	ማ	ma	ሜ	mie	ም	m	ሞ	mo
ሠ	se	ሡ	su	ሢ	si	ሣ	sa	ሤ	sie	ሥ	s	ሦ	so
ረ	re	ሩ	ru	ሪ	ri	ራ	ra	ሬ	rie	ር	r	ሮ	ro
ሰ	se	ሱ	su	ሲ	si	ሳ	sa	ሴ	sie	ስ	s	ሶ	so
ሸ	Se	ሹ	Su	ሺ	Si	ሻ	Sa	ሼ	Sie	ሽ	S	ሾ	So
ቀ	qe	ቁ	qu	ቂ	qi	ቃ	qa	ቄ	qie	ቅ	q	ቆ	qo
ቐ	Qe	ቑ	Qu	ቒ	Qi	ቓ	Qa	ቔ	Qie	ቕ	Q	ቖ	Qo
በ	be	ቡ	Bu	ቢ	bi	ባ	ba	ቤ	bie	ብ	b	ቦ	bo
ቨ	ve	ቩ	vu	ቪ	vi	ቫ	va	ቬ	vie	ቭ	v	ቮ	vo
ተ	te	ቱ	tu	ቲ	ti	ታ	ta	ቴ	tie	ት	t	ቶ	to
ቸ	ce	ቹ	cu	ቺ	ci	ቻ	ca	ቼ	cie	ች	c	ቾ	co
ኸ	Ke	ኹ	Ku	ኺ	Ki	ኻ	Ka	ኼ	Kie	ኽ	K	ኾ	Ko
ኀ	he	ኁ	hu	ኂ	hi	ኃ	ha	ኄ	hie	ኅ	h	ኆ	ho
ነ	ne	ኑ	nu	ኒ	ni	ና	Na	ኔ	nie	ን	n	ኖ	no
ኘ	Ne	ኙ	Nu	ኚ	Ni	ኛ	Na	ኜ	Nie	ኝ	N	ኞ	No
አ	a	ኡ	u	ኢ	i	ኣ	a	ኤ	ie	እ	A	ኦ	o
ከ	ke	ኩ	ku	ኪ	ki	ካ	ka	ኬ	kie	ክ	k	ኮ	ko
ወ	we	ዉ	wu	ዊ	wi	ዋ	wa	ዌ	wie	ው	w	ዎ	wo
ዐ	Oe	ዑ	Ou	ዒ	Oi	ዓ	Oa	ዔ	Oie	ዕ	O	ዖ	O
ዘ	ze	ዙ	zu	ዚ	zi	ዛ	za	ዜ	zie	ዝ	z	ዞ	zo
ዠ	Ze	ዡ	Zu	ዢ	Zi	ዣ	Za	ዤ	Zie	ዥ	Z	ዦ	Zo
የ	ye	ዩ	yu	ዪ	yi	ያ	ya	ዬ	yie	ይ	y	ዮ	yo
ደ	de	ዱ	du	ዲ	di	ዳ	da	ዴ	die	ድ	d	ዶ	do
ጀ	je	ጁ	Ju	ጂ	ji	ጃ	Ja	ጄ	jie	ጅ	j	ጆ	jo
ገ	ge	ጉ	Gu	ጊ	gi	ጋ	Ga	ጌ	gie	ግ	g	ጎ	go
ጠ	Te	ጡ	Tu	ጢ	Ti	ጣ	Ta	ጤ	Tie	ጥ	T	ጦ	To
ጨ	Ce	ጩ	Cu	ጪ	Ci	ጫ	Ca	ጬ	Cie	ጭ	C	ጮ	Co
ጰ	Pe	ጱ	Pu	ጲ	Pi	ጳ	Pa	ጴ	Pie	ጵ	P	ጶ	Po
ፀ	Xe	ፁ	Xu	ፂ	Xi	ፃ	Xa	ፄ	Xie	ፅ	X	ፆ	Xo
ጸ	xe	ጹ	xu	ጺ	xi	ጻ	xa	ጼ	xie	ጽ	x	ጾ	xo
ፈ	fe	ፉ	fu	ፊ	fi	ፋ	fa	ፌ	fie	ፍ	f	ፎ	fo
ፐ	Pe	ፑ	Pu	ፒ	Pi	ፓ	Pa	ፔ	Pie	ፕ	P	ፖ	Po
።	.	፣	,	፤	;	፥	:	?	!				

Declaration

I, the undersigned, declare that this thesis is my original work and has not been presented for a degree in any other university, and that all source of materials used for the thesis have been dulyacknowledged.

Declared by:

Name: Mulugeta AtsbahaSium

Signature: _____

Date: _____

Confirmed by advisor:

Name: Solomon Teferra (PhD)

Signature: _____

Date: _____

Place and date of submission: Addis Ababa, October, 2016.